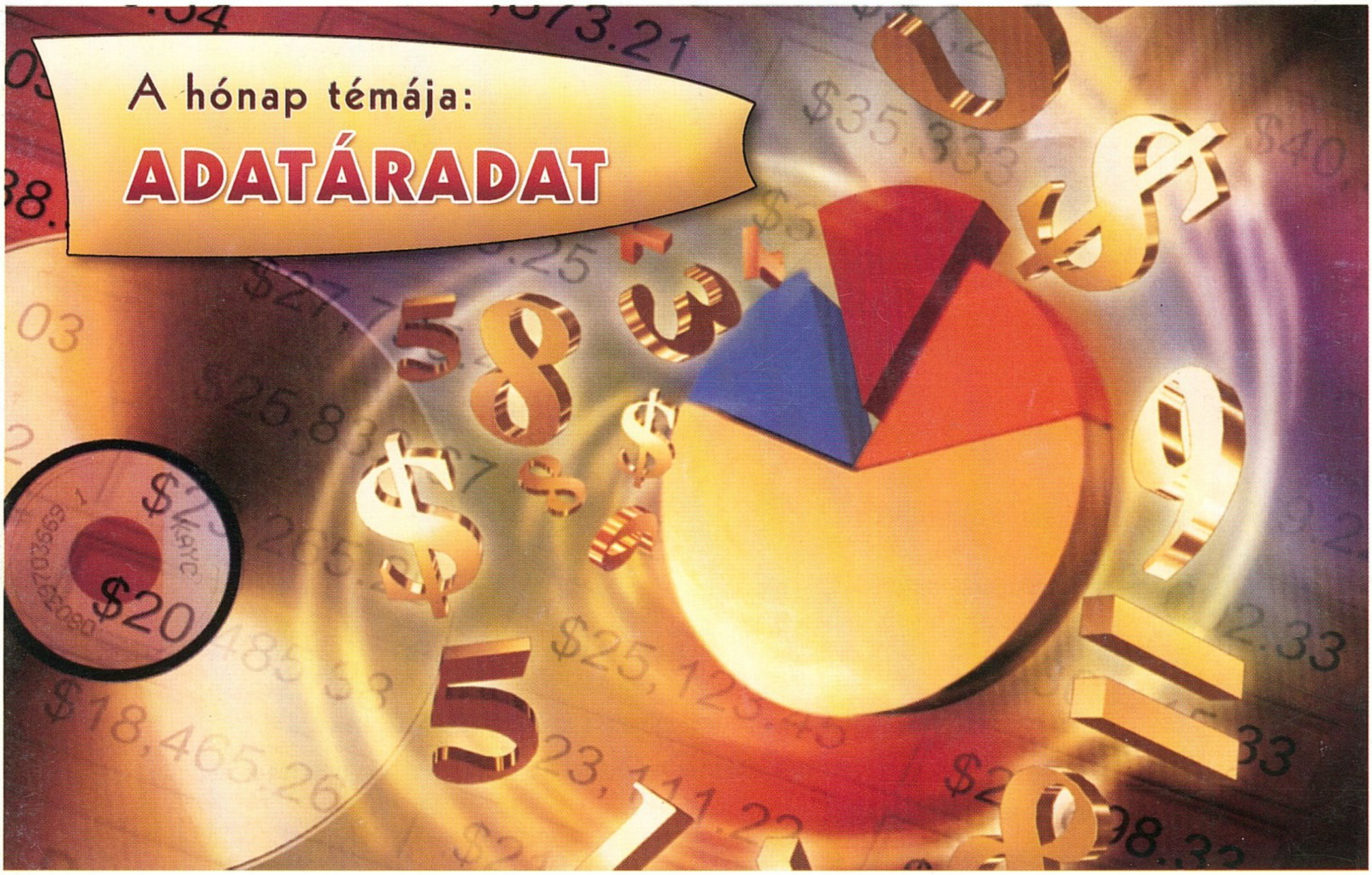
HÁLÓZAT * HARDVER * PROGRAMOZÁS * SZOFTVER
Ára 896 Ft

XIX. évfolyam 4. szám
2001. április

ÚJ ALAPLAP



A hónap témája:

ADATÁRADAT

CD-írók tesztje
Próbapad

Szelídítsünk pingvint
Programozástechnika

Platformfüggetlen vírusok
Vírusőrjárat

Most már digitálisan írunk alá?
Hálózat

Feladványok és adatbázisok
Kaleidoszkóp

9 771217 759005 01004



...internetből is

Aki megszokta már, hogy mindenből mindig a csúcsminőséget választja, az az internet esetében sem dönthet másképp. Az ő dolgukat könnyíti meg cégünk, a PSINet, mert...

- **a világ legnagyobb független kereskedelmi internetszolgáltatója vagyunk**
- **a legkorszerűbb technikai megoldásokat alkalmazzuk**
- **az üzleti szférára fókuszálunk**
- **az egész világon jelen vagyunk**
- **innovatív termékeket és szolgáltatásokat kínálunk**
- **képzett szakembergárdával állunk ügyfeleink rendelkezésére**
- **több mint 10 éves piaci tapasztalattal rendelkezünk**

Az eddig sem volt kérdés, hogy legyen-e internet elérhetősége. Most már az sem, hogy kitől...

....ha mindenből a csúcsminőséget szeretné.

PSINet® A világon minden elérhető.

PSINet Magyarország Kft. Tel.: 465-7859 Fax: 465-7899 Budapest 1134 Váci út 37., sales@psinet.hu, www.psinet.hu

ÚJ ALAPLAP

A Mikroszámítógép Magazin és az Alaplap hagyományait folytató magyar számítástechnikai folyóirat

Megjelenik havonta, CD-melléklettel

Főszerkesztő:
Faklen Pál

Szerkesztő:
Jakab Ágnes

A szerkesztőbizottság tagjai:
Aszalós László, Bánó György, Feleki Zoltán, Galántai Zoltán, Herczeg József, Kádár Zsolt, Kovács Attila, Mákos András, Nagy Tamás, Pogány Csaba, Sándor Gábor, Simay Endre István, Szappanos Gábor, Szondi Egon János, Vargha Dénes, Vékony Tamás

Szerkesztőség és kiadó:
1539 Budapest, Pf. 571
VI., Dózsa György út 84/b
Telefon: 322-4417, 322-5238
Fax: 351-8015
E-mail: alaplap@mail.datanet.hu

Weblap: http://www.alaplap.hu

Felelős kiadó:
Faklen Pál

Terjesztés:
Megyes Zsuzsanna

Hirdetésszervezés:
Árvai Katalin,
Galyasi Hedvig,
Tóth Zsuzsanna

Külföldi hirdetések:
PubliciTeam
Reklám- és Médiaügynökség
1537 Budapest I., Márvány u. 17.
Telefon: 356-1182 Fax: 214-9490

A kiadó a hirdetések tartalmáért és a nyomdakészen kapott hirdetések formájáért (és helyesírásáért) nem vállal felelősséget

Példányszámadatok hitelesítése:
Magyar Terjesztésellenőrző Szövetség
MATESZ

Ez a szám
8500 példányban jelent meg

Nyomtatás:
Zalai Nyomda Rt, Zalaegerszeg
Felelős vezető:
Czirkl György vezérigazgató

Terjeszti:
a Lapker Rt, a Hírker Rt, az NH Rt, az MP Rt LHI és számos számítástechnikai szaküzlet

Előfizethető a kiadónál:
Új Alaplap Kiadói Kft,
1539 Budapest, Pf. 571
Bankszámlaszám:
OTP 11706016-20788599

A lap példányonkénti ára: 896 Ft
Évi előfizetési díj: 8960 Ft

Külföldi előfizetés díja:
8960 Ft + postázási költség

HU ISSN 1217-7598

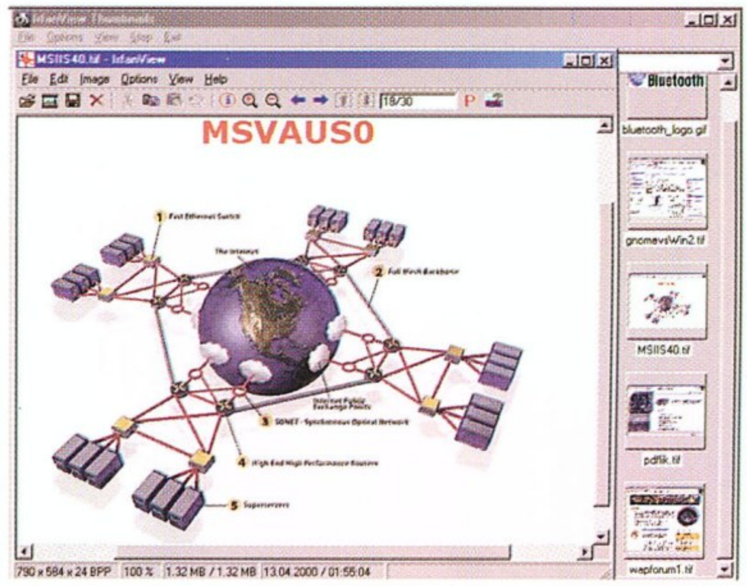
TARTALOM

XIX. ÉVFOLYAM 4. SZÁM, 2001. ÁPRILIS





Címlapképünk a Matrox reklámjából



FOKUSZ

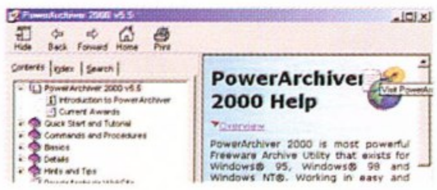
Fókuszban az adatbáziskezelés

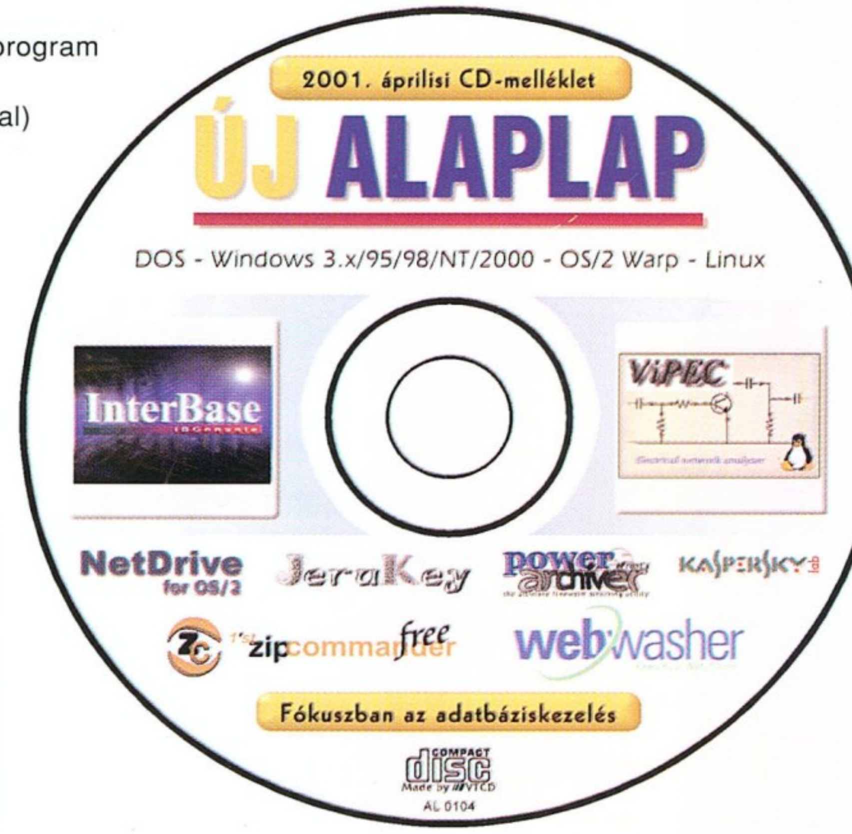
DBLIN	Adatbázisok kezelését segítő programok Linuxra
DBWIN	Adatbázisok kezelését segítő programok Windowsra
FDBS	Catalog of Free Database Systems, szabad forráskódú adatbáziskezelő rendszerek katalógusa (David Muir Sharnoff)
IB601	InterBase v6.01, adatbáziskezelő rendszer különböző platformokra
MYSQL	MySQL 3.23.36, adatbáziskezelő rendszer különböző platformokra
POSTGRES	PostgreSQL 7.1, adatbáziskezelő rendszer

LAPFORG

Lapraforgó

DIGALAIR	Most már digitálisan írunk alá? (Hálózat, 54. oldal)
	Az elektronikus aláírásról szóló törvényjavaslat általános vitája (Jegyzőkönyv, 2001. március 26.)
	Az elektronikus aláírásról szóló T/3847 számú törvényjavaslat (2001. február)
	Tervezet az elektronikus aláírás szabályozásának koncepciójáról (2000. április)
JAVA	Illusztrációk a Java tanfolyamhoz (Programozástechnika, 68. oldal)
KALEID	Feladványok és adatbázisok (Kaleidoszkóp, 63. oldal)
	SuperPro 3.4, sakkadatbázis demó (Lovass László)
OS2	Nemcsak OS/2 ... (Alternatíva, 32. oldal)
	MultiDesk 0.1.8, az OS/2 többfelhasználós működését támogató program
	NetDrive, 1.0.8, 2.x beta 2, fájlrendszer OS/2-höz
	Tonigy v1.0, fájlrendszer (Audio CD IFS)
	UpdD 1.4, a Warp 4-es telepítő CD-k frissítését elvégző program
	WinWpi v0.1, WPI formátumú fájlokat kezelő program
SZOFTVER	Listák és címkék komponensből (Szoftverportéka, 57. oldal)
	List & Label 7.0, jelentés- és címkekészítő program
URL	A lapban hivatkozott URL címek jegyzéke
	Webkalauz (62. oldal)
VISSZA	Shakespeare vagy Bacon? (Visszacsatolás, 62. oldal)
	Weblaprészletek Francis Bacon alkotói tevékenységéről
WINLIN	Windowstól a nyílt forráskódig (Alternatíva, 30. oldal)
	Windows Smart Card Toolkit 1.1, segédeszköz intelligens kártyákon futó programok fejlesztéséhez
	Service Pack 1 for Smart Card Toolkit 1.1, javítócsomag a Smart Card fejlesztőkészlethez





SZERSZAM

Szerszámosláda

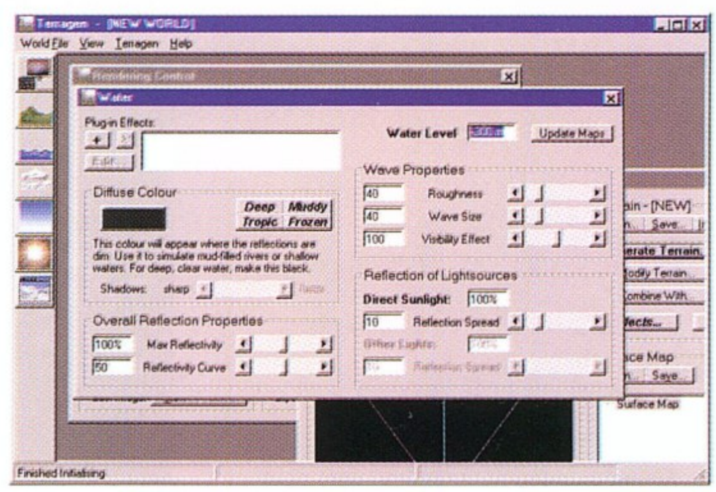
DOS	DOS-os alkalmazások
FESZER	Gyakran szükséges programok
LINUX	Linuxos alkalmazások
OS/2	OS/2-es alkalmazások
VIRUS	Vírusirtók
WIN3X	16 bites windowsos alkalmazások
WIN9X	32 bites windowsos alkalmazások

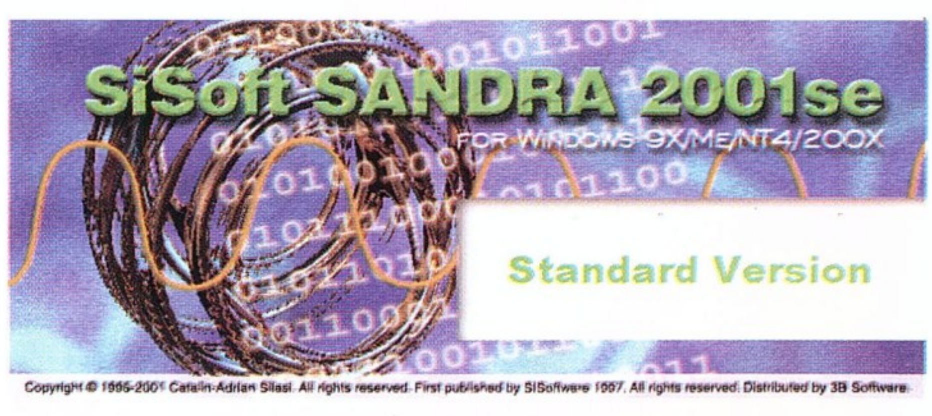


VENDEG

Vendégoldal

CDM	The Rosetta Stone, nyelvoktató program, English 4 (CDM – Europress Hungary)
DECROS	Protect for Windows 3, adatbiztonsági szoftver Windowsra (Decros)
DELCOMP	Delphi komponensek
GACSFALV	Gácsfalvy László programjai
	Raktár v1.0, raktári nyilvántartó és számlázó rendszer
	Angol-magyar szótár (Access 97-es és Access 2000-es változat)
GNOME	Gnome 1.4, grafikus felület Unixra
KASPERSK	Kaspersky Anti-Virus (AVP) Lite
	Kaspersky Anti-Virus (AVP) for Linux Workstation 3.0.135
MICROSFT	Internet Explorer 6 Public Preview (fejlesztés alatt álló változat)
	Javítócsomagok a Microsfttól
OPENOFF	OpenOffice 6.0 build 625, irodai programcsomag, fejlesztés alatt álló változat (forráskód és bináris állományok)
PROFIMED	MiniTavasz demó (Profi-Média Kft)
SZOFTABC	A Szoftver ABC honlapja
WEBWASH	WebWasher 3.0, reklámablakokat kiszűrő program különböző platformokra
ZIPCOMM	1'st ZipCommander, magyar fejlesztésű tömörítő, fájlkezelő és internetböngésző





JATEK

Játékvár

ARENA	ArenaBall 0.1.0, kézilabda játék
ROCKS	Rocks 'n' Diamonds v2.0, gyűjtögető játék
STARTRON	StarTron 1.0, ügyességi játék

Adatbázis-korszakváltás

Sajátos viszonyban vagyunk a számszerűsíthető adatokkal. Amikor sikerül szert tennünk egy kellőképpen áttekinthetetlen adathalmazra, akkor megpróbáljuk rendszerezni. Utána már ebben a rendezett formában folytatjuk az adatfelhalmozást, de előbb-utóbb ismét eljutunk a rendezetlenség állapotáig, és akkor kezdődik az egész elölről. A folyamat sok szempontból olyan, mintha a sivatagban ide-oda lapátolnánk a homokdombokat. Hogy mégse így történjen, legalább az adatok újbóli beírását, rögzítését célszerű elkerülni. Ebben segítenek nekünk az adatbázisok és az adatbázisokat kezelő alkalmazások.

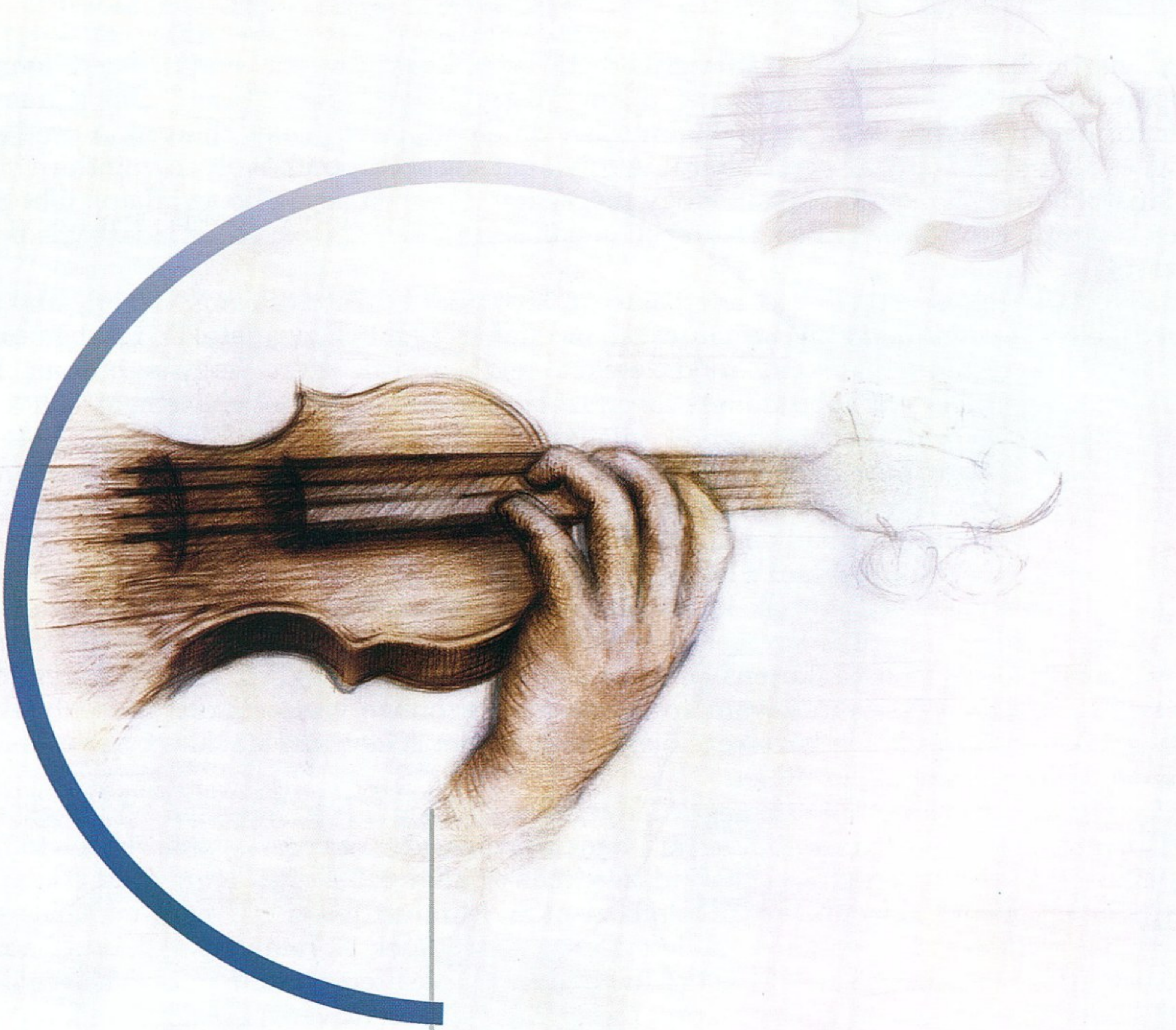
Az adatbázist kezelő szoftver és az adatbázisban lévő adat között is sajátos viszony alakult ki. Az adatmenedzselő programok egyéni formátumban tárolták és kezelték az adatokat, részben ésszerűségi okokból, hiszen keresték a hatékony módszereket, részben csak azért, hogy a hasonló funkciójú konkurensektől valamiben különbözzenek. Ez az izoláltság viszonylag jól elviselhető volt kezdetben, a központi gépek köré szervezett rendszerek világában, sőt a PC-s adatfeldolgozás első időszakában is, amikor a PC-s hálózat még nem ért túl a vállalat kapuján. A szigetrendszerek azonban ma már fokozatosan háttérbe szorulnak, és inkább csak az otthoni vagy kisirodai adatbáziskezelésben élnek tovább. A tipikus vállalati informatikai környezet egyre inkább megváltozik.

Napjainkra elszigetelődött például a legtöbb xBase alapú rendszer. A nagy karriert befutott, DOS-os dBase III és III+ ma már inkább csak kuriózum, és a Borland által a dBase nevű cégnek 1999-ben licencbe adott Visual dBase-ről is keveset hallunk, bár 2000 végén megjelentették a dB2k programot. Ugyanilyen helyzetben van a konkurens FoxBase alapján fejlesztett FoxPro, amelynek verziókövetése a Visual Studio csomagban ugyan folyamatos, használata azonban visszaszorult a kisebb feladatokra. Ugyanígy a Microsoft másik adatbáziskezelőjére, az Accessre is kevesen bíznák rá egy többmilliárdos forgalmú cég üzleti adatainak kezelését.

Az informatikai környezetváltozás egyik lényeges sajátossága, hogy gyorsan bővülnek a vállalati hálózatok, és már nemcsak a kapun „bújnak ki", hanem a városokon, az országhatárokon, a földrészeken is átnyúlnak. Az egyre kiterjedtebbé váló rendszerekben az adatok belterjes gyarapodása is öngerjesztő folyamat, ami egyre több feladatot ró az adatbázisokra. Mindezt tetézi egy nem mellékes szerkezeti változás: a hagyományos számadatok mellett a szövegek, képek és hanganyagok is az adatbázisok természetes részévé válnak. A változások eredményeként ugrásszerűen több információ fizikai tárolásáról kell adatbázisrendszerben gondoskodni, és ehhez a külön hardveres eszközök fejlesztése mellett meg kell oldani a szoftveres problémákat is. A korábban használt rendszerek sokszor használhatatlanná válnak az új környezetben. Ha például egy xBase alapú rendszerre gondolunk, nem lenne gazdaságos a több MB-os adatbázisokhoz szükséges, és ugyancsak több MB-os indexállományoknak is helyet szorítani. Egyetlen nagy fájl pedig biztonsági és elérhetőségi megfontolásokból nem a legjobb választás.

Ennek az átalakuló — vagy részben már át is alakult — adatáradatnak egyes szegmenseit szeretnénk felvillantani mostani összeállításunkban. Témaválasztásunk a nagy adatkezelő rendszerekre koncentrál, de ez már sokkal közelebb van a „kis adatbázisok" világához szokott szakemberekhez is, semhogy közömbösen elmenjenek mellette.





Egy lassan beérő gondolat

Az IMS-től a relációs adatbázisokig

Ma már alig emlékszünk rá, mekkora küzdelmet folytattak egymással az adatbázis-fejlesztő cégek az elsőbbségért — és hogy miként bontakozott ki ebből a harcból egy lényegében egységes, egyirányú fejlődési tendencia. Érdemes végiggondolni, mi volt a belső logikája ennek a fejlődésnek, és miként lett az adatbáziskezelés a számítástechnika egyik elismerten legfontosabb alkalmazási területe.

Kezdetben minden programnak magának kellett gondoskodnia azokról az adatokról (bevitelükről, tárolásukról, felhasználásuk módjáról), amelyekkel dolgoztak. Hamar világossá vált, hogy nem szabad belekeverni az adatokat a programba, mert akkor a tartalmi és formai változások követése igen nagy gondot okozhat. Egy elvileg is súlyosabb probléma pedig akkor merült fel, amikor több program, sőt több programozó akarta használni ugyanazokat az adatokat. Ebből az egyre gyakrabban előálló helyzetből értették meg a programozók, hogy mennyire fontos az adatkarbantartás szabályainak pontos meghatározása és betartása, és egyáltalán az adatok hozzáférési felületének precíz definiálása.

Az első sikerek

Eleinte külön karbantartó programok készültek minden programhoz. Később azonban a programozók már szerettek volna megszabadulni ettől a favágás jellegű munkától, és joggal elvárták, hogy ez a teher kerüljön le végre a vállukról. Ahogyan a fájlkezelési feladatok és más rutintevékenységek elvégzésének a felelősségét át tudták vállalni az operációs rendszerek, ugyanúgy központilag elintézendő feladatnak tűnt az adatkezelés rutintevékenységeinek intézése is. Miért ne lehetne olyasféle programrendszereket kifejleszteni, amelyek akár bonyolultabb adatstruktúrák kezeléséről is gondoskodni tudnak?

Az első igazi adatbáziskezelő rendszerrel az IBM rukkolt ki 1969-ben. A Nagy Kék „Information Management System" (IMS) nevű rendszerével rövid idő alatt jó üzletet csinált, és általános elismerést aratott. Nagyon egyszerű megoldást használt az adatok között fennálló kapcsolatok leírására: közönséges fastruktúrákat alakított ki az adatokból. Lehetetlen észre nem vennünk ezeknek a struktúráknak a hasonlóságát a DOS könyvtárszerkezetével: az IMS olyan formában ábrázolja az adatstruktúrákat, ahogyan a könyvtárrendszerek elhelyezik a fájlokat a háttértárolón. Az IMS sikerét éppen ennek az egyszerűségének köszönhette, hiszen az ilyen szerkezet jól áttekinthető, és a hatékony fizikai megvalósítás sem okoz problémát.

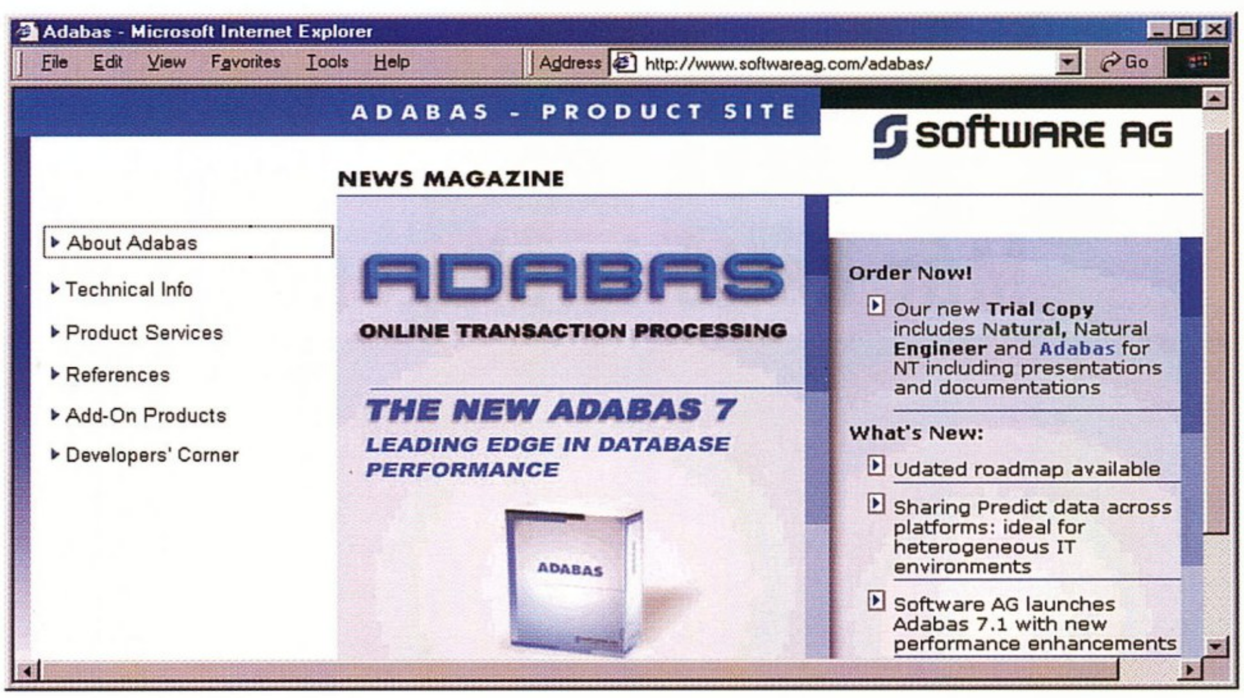
A sikerért természetesen meg is kellett dolgozni: az IBM kényelmes, szabványosított eljáráskészletet készített az IMS-hez. A programozás kellemesebbé tétele érdekében pedig az eljárások meghívása helyett hamarosan makrókat kezdett alkalmazni. A makrók kifejtését rá lehetett bízni egy előfordítóra (ez volt a DL/1), amely azután korrekt Cobol vagy PL/1 utasítások formájában helyezte el a programban a megfelelő utasításokat. A program készítője tehát megszabadult egy olyan másodrangú (de rengeteg hibalehetőséget hordozó) tevékenységtől, mint az adatok kezelése, és teljes figyelmét az adatbázisból elővett adatok feldolgozására koncentrálhatta.

A Codasyl-kitérő

Az adatbázis-problematika fejlődésében a következő lépést a nemzetközi Codasyl szabvány megjelenése és az ezt kielégítő IDMS (Integrated Database Management System) rendszer kifejlesztése jelentette. A Codasyl koncepciója kétségtelenül tökéletesebb volt, mint az IMS-é: a Codasyl (1) a fák helyett megengedte hálók alkalmazását, (2) elválasztotta az adatdefiníciókat az adatkezeléstől, és (3) jelentősen javította az adatbiztonságot, mert megfelelő definíciókkal elegánsan meg lehetett oldani az adatok elrejtését.

Maga a helyes elv azonban nem sokat ér, ha gyatra a megvalósítás. Sok programozó emlékezhet még rá, hogy egy ideig nálunk is divat volt az adatbázisok tervezése, és hatalmas pénzekért sok magyar cég is megvásárolta a nehézkes IDMS monstrumot. Az alkalmazási próbálkozások azonban legtöbbször keserű kiábrándulással jártak, a drágán beszerzett programcsomag nagyon kevés cégnél vált be. Csak később derült ki, hogy akkoriban már létezett a Codasyl irányelveinek sokkal ügyesebb megvalósítása is, az Adabas rendszer. Ezt viszont még sokáig nem adták el Keletre…

Körülbelül a 70-es évek elejétől szinte az egész számítástechnika központi kérdésévé nőtte ki magát az adatbázisok



fejlesztése. Folytak az elméleti kutatások, és rengetegen keresték a hatékony gyakorlati megoldásokat. Két irányban volt különösen intenzív a kutatás. Az egyik irányzat azt a felismerést igyekezett hasznosítani, hogy az adatdefiníciók éppúgy felfoghatók adatoknak, mint maguk a hagyományos értelemben vett adatok, vagyis az adatleírások is kezelhetők adatbázisban.

Kikristályosodott az adatok absztrakciójának egy magasabb szintje, és megjelent a metaadatbázisok fogalma. A hétköznapok gyakorlatában ez úgy realizálódott, hogy felbukkantak az adatkatalógusok mint speciális adatbázisok, és az új, modernebb szemléletű adatbázisok ezekre épülve készültek el. Az elmélet tehát átszűrődött a gyakorlatba: a tervezők ráébredtek, hogy mindkét adatbázisféléhez ugyanazokat a szoftveres eszközöket lehet felhasználni.

Heuréka! Vagy mégsem?

A másik fejlődési irányzat nem az adatbázisok statikus leírásával foglalkozott, hanem inkább az adatok kezelésének dinamikájával. E. F. Codd alapvető kutatásai nyomán megindult az adatbázisok „algebraizálódása". Codd megalkotta a relációs adatbázismodellt és a hozzá csatlakozó relációs kalkulust. Kimutatta, hogy az adatbázisok adatain végzett mindenféle művelet halmazműveletnek tekinthető, és leírható az algebrai nyelvhez nagyon közelálló relációs algebrai nyelven.

Codd meglátása azért tekinthető korszakalkotónak, mert így „szárazon", az adatokhoz való hozzáférés nélkül elemezni lehet a kifejezések korrektségét, sőt az előzetes optimalizáció is megejthető. Elvileg tetszőleges bonyolultságú műveletsorozat elvégezhető az adatok halmazán, bár a méretek és a számítási idő legfeljebb hozzávetőlegesen jósolható meg.

A Codd elvein alapuló rendszer kidolgozása eleinte szinte megvalósíthatatlannak látszott. Főleg a hatékonyságával kapcsolatban merültek fel aggályok, azok pedig az egész koncepciót illuzórikussá tették. Kisméretű kísérleti rendszerek születtek ugyan (még olyanok is, amelyek természetes nyelvű interfésszel működtek), de ezek inkább csak az ötlet szemléltetésére voltak jók. Mivel reális méretek esetén e rendszerek hatékonysága jóval elmaradt a hierarchikus vagy más elven működő rendszerekétől, a gyakorlatias gondolkodású szakemberek nem is fordítottak rá nagyobb figyelmet.

Az IBM maga is kétlelkű maradt ebben a sorsdöntő kérdésben. Fejlett szoftvertermékeivel kétségtelenül jó hírnevet szerzett a hagyományos adatbázispiacon. Legjobb kutatói azonban kitartottak eredeti elképzeléseik mellett, és gyökeres változtatásra készültek. Codd, az ő gondolatait továbbfejlesztő amerikai-kínai P. S. Chen és a zseniális adatbázisszakértő C. J. Date kidolgozta egy valóságos „adatbázisgép" részletes tervét, az ún. Database R rendszert, mely a relációs elveken alapult, kommunikációra pedig a kísérleti Sequel rendszer fejlettebb változatát, az SQL nyelvet használta.

Az SQL és a versenytársak

Az „R rendszer" koncepció kidolgozóinak lényegében az SQL nyelvvel sikerült legyőzniük az IBM menedzsmentjének ellenállását. A fordulópontot az jelentette, hogy igazolni tudták: az SQL nyelvre épülő interfészt akkor is érdemes elkészíteni, ha az adatbázis fizikai megvalósítása a hagyományos módszerekkel történik, mert a relációs szemléletű megközelítés a felhasználók számára sokkal világosabbá teszi a problémák átlátását. Az SQL-t mint humán interfészt még akkor sem kell megváltoztatni, ha a relációs elvek megvalósítására később valami modernebbre cserélik ki az adatbázisgépet.

A szellemet már nem lehetett visszaparancsolni a palackba. Az IBM nem siette el a dolgot, és a maga nehézkességével fogott munkához. Olyannyira, hogy mire kifejlesztette saját rendszerét, a DB2-t, a relációs adatbázisok fejlesztésében sorra megszülettek és megerősödtek a versenytársak is, és talán nem véletlen, hogy mindegyik az SQL valamilyen dialektusával lépett fel. Az IBM-ből kivált Oracle fejlesztőgárda egyhamar az élre került új termékével, az Oracle adatbázissal. A Relational megalkotta az Informixet, és ebben megvalósította a logikai szinten végrehajtható optimalizálást. A Novell XQL rendszere egységes fizikai adatkezelést biztosított a SoftCraft felvásárlásával megszerzett Btrieve rekordkezelő rendszerrel, a felszínen azonban hű maradt az SQL-szerű felülethez.

Külön érdemes szólni a Sybase nevű fejlesztőtársaságról, amely két jelentős újítással szerzett magának jó hírnevet.

1. Lényegesen növelte az adatkezelés biztonságát azzal, hogy bevezette a kétfázisú tranzakciókezelést, így mindaddig nem válnak érvényessé a változtatások a rendszerben, míg a tranzakció minden lépése hibátlanul le nem fut.

2. Elsőként alkalmazta a kilens-szerver elvet, ezzel szilárdan megvetve a lábát a PC-hálózatok és a Windows világában.

Hogy mennyire a dinamikus és határozott üzletpolitikán múlt ebben az időben a cégek gyors felemelkedése, az két jelentős döntésen mérhető le a legjobban. Az egyik az volt, amikor a Sybase magához csábította és „kivásárolta" a PowerSoft kiváló fejlesztőgárdáját.

Másik fontos üzleti döntésüknek az bizonyult, hogy újításaik birtokában sem ragaszkodtak monopolhelyzetükhöz: megalkudtak a Microsofttal, és rendszerüket átengedték neki továbbfejlesztésre. Ebből nőtt ki a „Microsoft SQL Server" elnevezésű rendszer. Végső soron ezzel segítette hozzá a Sybase a Microsoftot, hogy a vállalati szférában szoftver nagyhatalommá váljék...

Mit hoz a holnap?

Az adatbázisok fejlődésének következő nagy fordulatát az objektumorientált szoftverfejlesztési koncepció elsöprő győzelme hozta meg. A várakozások ellenére ez mégsem ingatta meg az adatkezelésben a relációs adatbázisok uralmát, sőt az SQL pozíciói nem gyöngültek az adatkeresésben. (Az OOP által előidézett változások taglalásától most eltekinthetünk, bőségesen foglalkoztunk ezzel a témával 1988. májusi számunkban.)

Egyelőre csak találgathatjuk, hogy lesz-e újabb nagy fordulat az adatbázisok fejlődésében. Vajon véglegesen győzött-e az SQL, vagy találnak még jobbat? És mi történik közben a mélyben? Hoz-e valami alapvetően újat az objektumorientált szoftverfejlesztési koncepció? Az osztott rendszerek bevezetésével vajon sikerül-e nagyságrendekkel javítani a rendszerek hatékonyságát?

Minden lehetséges. Attól mindenesetre óvakodjunk, hogy véglegesen lezártnak tekintsük az adatbázisok problematikáját. Annyira összefonódnak ezen a területen a hatékonyságnak, az adatkezelés biztonságának és a kellemes felhasználói környezet kialakításának kérdései, hogy újabb eredeti ötletek megvalósításával ismét a feje tetejére állhat minden.

Vargha Dénes

Kapcsolódó cikkek az Új Alaplap korábbi számaiban:

V. Nagy Edit: Adat(bázis)centrikusan. 1995/12, pp. 58-60.

Németh Miklós: OOP adatbázisok. 1998/6, pp. 15-16.

Nick János: Jobbak, mint a relációsak. 1998/6, pp. 17-18.

Németh Miklós: Sokszínűség. 1998/6, pp. 12-13.

Üzleti intelligenciaforrás

Microsoft SQL Server 2000 platform

Az üzleti előny általában nem az adatok birtoklásából, hanem azok kellő ismeretéből és felhasználási módjából származik. A vállalkozások és más szervezetek nagy mennyiségű adatot gyűjtenek össze, ami többnyire a napi működést támogató operatív adatbázisokban és archivált formáikban találhatók meg. Az üzleti informatika feladata, hogy ezek alapján információkat szolgáltasson a cég vezetése és az elemzők számára. Ehhez kell az operatív adatbázisokból származó adattömeget egységes formában tárolni, a nem számítástechnikai szakemberek számára is biztosítva a gyors adatlekérdezés eszközeit, és gondoskodva az elemzés során születő döntéseknek az üzleti folyamatba való eljuttatásáról. Az SQL Server 2000 többek között ennek az üzleti intelligenciának a platformja.

Az adatmennyiségtől és a gazdálkodó szervezet korábbi beruházásaitól függően az operatív adatok lehetnek egyszerű indexszekvenciális fájlokban, táblázatkezelőkben, relációs adatbázisokban vagy különböző nagygépes rendszereken. A nagyobb cégek nem ritkán két-három adatbáziskezelő rendszert és ennél is több — logikai vagy fizikai felépítésükben eltérő — adatbázist használnak. Az adatraktár szerkezete viszont alapvetően eltér az erősen normalizált (kötött) tranzakciókezelő adatbázisok szerkezetétől. A tranzakciós adatbázisokat ugyanis az adatkezelési sebességre optimalizálják, az adatraktárt pedig a felhasználók szemszögéből építik fel. Az adatraktár szerkezete ezért többnyire csillag vagy hópehely sémát mutat. A központi ténytábla körül az elemzési szempontok szerint összeállított dimenziótáblák találhatók. Az 1. ábra olyan adatbázist mutat be, ahol az eladásokat idő, bolt, ügyfél és termék szerint lehet vizsgálni.

Akár a vállalat egészére vonatkozó adattárházat, akár egy kisebb szervezeti egység adatait összefogó adatraktárt alakítunk ki, meg kell oldanunk a heterogén forrásokból származó adatok kinyerését, egységes formára alakítását, ellenőrzését, és az adatraktár tábláiba történő betöltését. Az SQL Server 2000 Data Transformation Services (DTS) szolgáltatása (elsősorban) ezt végzi el. A DTS bemenete és kimenete lehet bármilyen OLE DB vagy ODBC felületen keresztül elérhető adatforrás, például MS Access, Excel, dBase, Paradox adatbázis vagy csak sima, de tagolt szövegfájl. A DTS az így beolvasott adatokon bizonyos átalakításokat el tud végezni. Az átalakítások az egyszerű konverzióktól a szkript nyelveken (VBscript, Jscript, Perl stb.) megírt eljárásokon át a tetszőleges nyelven megírható ActiveX komponensek által végzett transzformációkig terjedhetnek. A fontosabb DTS feladatok listáját az 1. táblázat tartalmazza.

A 2. ábrán egy tipikus DTS csomag látható. A táblák inicializálása után öt adatforrásból transzformáljuk át a csillag séma dimenziótábláit a Northwind_OLAP adatbázisba. (A transzformációkat a szürke nyilak jelzik.) Ezután a központi ténytábla feltöltése következik az NW-Sales_Fact adatforrásból, végül — ha az előző lépés sikeres volt — egy OLAP adatkocka feldolgozása történik meg.

Az adatok elemzése

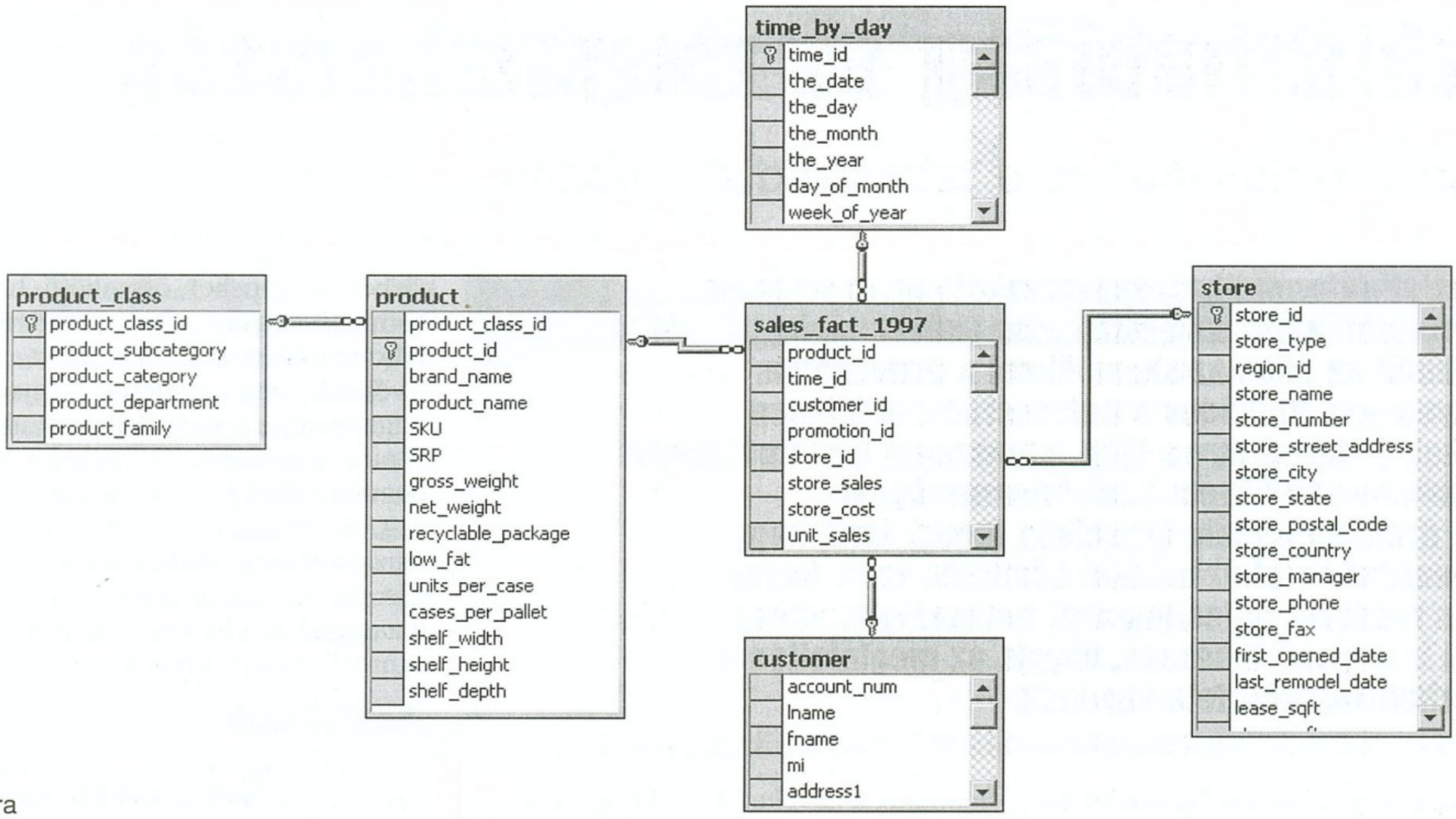
Az adatraktár adatait közvetlenül, SQL parancsokkal, illetve jelentéskészítő eszközökkel kérdezhetjük le. Nagyméretű táblák és akár több terabájtnyi adat esetén ezek az SQL lekérdezések azonban hosszú időt vehetnek igénybe, bármilyen kiváló relációs adatbáziskezelőt és erős számítógépet használunk hozzá. A gyakori „szabványos" lekérdezések felgyorsítására szolgálnak az SQL Server 2000-ben az ún. indexelt nézetek. A nézet (view) indexelése a nézet tartalmát „materializálja", eltárolja. Így azok a lekérdezések, amelyeket az indexelt nézet tartalma lefed, lényegesen gyorsabban hajthatók végre, mintha az adatbáziskezelőnek a nézet alatti táblákból kellene dolgoznia. A tetszőleges, ad hoc jellegű lekérdezésekre viszont nem tudunk indexeléssel felkészülni. Itt segít az OLAP.

OLAP adatkockák

Az OLAP az online analitikus feldolgozás angol rövidítése (On-Line Analitical Processing). Olyan eszközöket jelent, amelyek összesített (aggregált) adatok több szempont (dimenzió) szerinti gyors lekérdezését támogatják. Alapkövetelmény a rövid, legfeljebb egy-két másodperces válaszidő akkor is, ha a felhasználók előre nem látható, „tetszőleges" lekérdezéseket fogalmaznak meg. Az OLAP adatbázisok az adatokat n-dimenziós adatkockákban tárolják. A kocka dimenziói az elemzési szempontok: idő, termék, vásárló, terü-

1. táblázat

Megnevezés	Funkció
File Transfer Protocol Task	FTP letöltés
ActiveX Script Task	Szkript végrehajtása
Transform Data Task	Adatpumpa és transzformáció két adatforrás között
Execute Process Task	Tetszőleges alkalmazás végrehajthatása
Execute SQL Task	SQL parancsok végrehajtása
Data Driven Query Task	A bemenő adatok által vezérelt lekérdezések végrehajtása
Copy SQL Server Objects Task	SQL Server objektumok másolása SQL Server adatbázisok között
Send Mail Task	Levélküldés
Bulk Insert Task	Gyors adatbetöltés SQL Serverbe
Execute Package Task	Beágyazott DTS csomag(ok) végrehajtása
Message Queue Task	MSMQ üzenetek kezelése
Analysis Services Processing Task	OLAP feldolgozás
Dynamic Properties Task	DTS csomag dinamikus paraméterezése
Data Mining Prediction Task	Adatbányászati előrejelzés



1. ábra

let stb. A kocka celláiban numerikus adatok találhatók: darab, eladási ár, költség stb.

Az OLAP rendszerek a válaszidőt többnyire úgy gyorsítják fel, hogy a lehetséges aggregátumokat előre kiszámolják. Ez a megoldás elvileg tökéletes, gyakorlatilag viszont az adatrobbanás néven ismert jelenséghez vezet: a tárolókapacitás iránti igény az elemzési szempontok exponenciális függvényeként növekszik. Az SQL Server 2000 Analysis Services mint OLAP-kiszolgáló az adatrobbanás problémáját intelligens, szelektív összesítő eljárással oldja meg, gyors, egy másodperc alatti válaszidőket biztosítva, miközben nem kell a tárolóhelyet a végtelenségig bővíteni. Az adatok tárolása nagyon hatékonyan, tömör szerkezetben történik, ezért az OLAP kocka mérete a nyers input adatoknak többnyire csak 20-40 százalékát teszi ki.

Ezt a hatékonyságot példázza a T3 projekt: http://www.microsoft.com/sql/techinfo/terabytecube.htm. A T3 adatkocka 1,2 terabájtnyi adatból készült. A ténytábla 7,7 milliárd sort tartalmazott. Az adatkocka mérete ugyanakkor „csak” 471 gigabájt, kevesebb, mint az input adatok 40%-a. A rendszer válaszideje 0,2–0,8 másodperc.

Az SQL Server 2000 Analysis Services lehetővé teszi, hogy az elemzést végző analitikus vagy döntéshozó az analízis során akciókat indítson, így azonnal visszacsatolódik a cég folyamataiba. Az akciótípusok listáját a 2. táblázat mutatja be.

Adatbányászat

Az OLAP kiváló megoldás adataink elemzésére, feltéve, hogy megvannak a dimenziók, az elemzés szempontjai. Más esetekben viszont nem nyilvánvaló, hogy egy adathalmaz milyen összefüggéseket rejt. Például ismerjük ügyfeleink adatait, az általuk vásárolt termékek és a termékeket árusító boltok jellemzőit, de nem tudjuk, hogy milyen tényezők befolyásolják a vásárlási döntést. Ilyen esetekben segíthetnek a statisztikai módszerek vagy az adatbányászati eljárások.

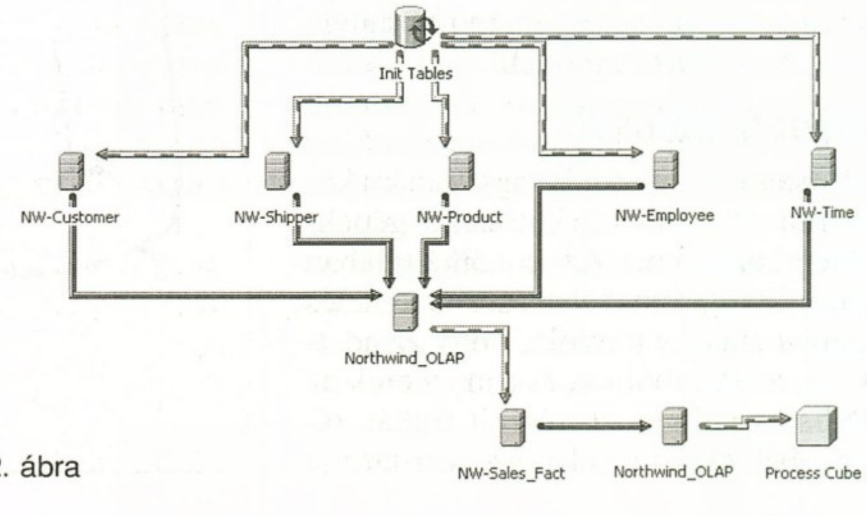
Az SQL Server 2000 Analysis Services nemcsak OLAP, hanem adatbányászati támogatást is nyújt. A relációs vagy OLAP adatok elemzését egy fürtözési algoritmus és egy döntési fa algoritmus segíti. (Az OLE DB for Datamining specifikáció lehetővé teszi külső gyártóktól származó további algoritmusok szabványos elérését is.) A 2. ábrán egy döntési fa látható. A döntési fa szerint a vásárlók döntését az adott termék esetében leginkább a választék határozza meg, ezt követi a vásárló életkora, és végül a jövedelem nagysága. Ha egy döntési fát és egy relációs táblát PREDICTION JOIN segítségével összekapcsolunk, a fenti adatok ismeretében előrejelzések készíthetők a vásárlási döntésekre vonatkozóan.

További információk találhatók az SQL Server 2000-ről és a termék üzleti intelligencia alkalmazásairól a www.microsoft.com/sql és a www.microsoft.com/sql/productinfo/dataware.htm címeken.

Koszó Károly
karolyk@microsoft.com

2. táblázat

Típus	Funkció
Command Line	Tetszőleges végrehajtható program indítása
Statement	OLE DB parancs végrehajtása
HTML	HTML szkript végrehajtása (megjelenítése) böngészőben
URL	Weblap megjelenítése
Data set	MDX lekérdezés
Rowset	SQL lekérdezés
Proprietary	Egyedi alkalmazás paraméterezése



2. ábra

Skálázhatóság és megbízhatóság

A „shared-nothing" és a „shared-disk" modell

A vállalati informatikai rendszerektől egyre nagyobb rugalmasságot és megbízhatóságot követelnek. Ezen belül az adatbáziskezelőknél a biztonságos és folyamatos működés a kulcskérdés, miközben meg kell birkózni egyre több felhasználó kiszolgálásával és a rohamosan növekvő adatmennyiséggel. Nincs azonban mindig lehetőség (pénz) arra, hogy a kapacitást beruházásokkal bővítsék, ezért fontos, hogy a rendszer teljesítményét technológiai váltás nélkül is növelni lehessen, vagyis az megfeleljen a skálázhatóság követelményének.

Amikor egy adott konfigurációt kinőttünk, és az általa nyújtott szolgáltatás teljesítménye már nem elegendő számunkra, a hardverbővítés a legegyszerűbb megoldás. Általában a rendszer valamelyik erőforrásának elégtelenségére utal a válaszidő növekedése is, és lehetséges, hogy a többi még elegendő lenne, ezért fontos pontosan tudnunk, hol van rendszerünkben a szűk keresztmetszet.

A nagyobb processzorteljesítmény eléréséhez további processzorokat helyezhetünk a gépbe (amennyiben lehetséges), vagy másik, többprocesszoros gépre cserélhetjük a jelenlegit. Ha a lemezkapacitás korlátjába ütközünk, akkor egy gyorsabb merevlemezes alrendszerrel megoldhatjuk a problémát. Ha a hálózat a szűk keresztmetszet, akkor berakhatunk a gépbe még egy hálózati kártyát. Ezekkel a lépésekkel orvosolni tudjuk a problémát... egy ideig. Mit tudunk viszont tenni, ha a hardvergyártó által szállítható legnagyobb gép is kevésnek bizonyul? Beszerzünk és üzembe helyezünk egy másik gépet is. A két gép teljesítményét ekkor azonban „össze kellene vonni". Adatbázis környezetben erre két alapvető megoldás van: a „shared-nothing" és a „shared-disk" modell.

„Shared-nothing"

A „shared-nothing" (vagyis „nem közösködünk") modell különálló gépekkel építkezik, amelyek között általában csak hálózati kapcsolat van. A terhelés megosztása úgy történik, hogy az adatbázist szétdaraboljuk, és a szerverek az adatbázis különböző részeit fogják tárolni. Példaként az első szerver tárolja mondjuk a neveket A-tól D-ig, a második E-től H-ig, és így tovább. Hogy kívülről az egész egyetlen egységként jelenjen meg, összetett nézetet használhatunk, amely összevont táblában láttatja a különböző adatbázisokban tárolt adatokat. Ha adatmanipulációs műveletet végzünk, akkor a nézethez kapcsolt triggerek (szerveroldali működtető programok) fogják a megfelelő adatbázisban végrehajtani a megfelelő műveletet.

Az egész folyamat lényege a „two phase commit", ami lehetővé teszi a tranzakció biztonságos végrehajtását úgy, hogy abban több adatbáziskezelő vesz részt. Az informatika egyik legbonyolultabb algoritmusáról van szó, mert úgy kell elvégezni a tranzakciót, hogy közben a gépeket összekötő hálózat bármikor megszakadhat. Az adatintegritás megőrzése érdekében ezért a résztvevőknek természetesen egymástól függetlenül is vissza kell tudniuk görgetni a tranzakciót. A végleges átadás (commit) pedig csak akkor történik meg, ha minden résztvevő igazolta a tranzakció megtörténtét. Ez persze egy második kapcsolatfelvételt is igényel, innen ered az elnevezés, a „two phase commit", azaz a kétlépéses átadás.

„Shared-disk"

A „shared-disk" vagy „lemezmegosztásos" modellben a gépek fizikailag közös merevlemezes alrendszert használnak (minimum háromdobozos az architektúra: 2 gép + 1 lemez). A merevlemezes alrendszerben lévő adatbázishoz bármelyik gép hozzáférhet. Ezt az architektúrát fürtözési (cluster) technikának hívjuk. Ha a gépek a közös merevlemezes alrendszert egyidejűleg tudják elérni, akkor használható az Oracle Parallel Server (OPS). Ha csak felváltva képesek az elérésre, akkor hibatűrő megoldásról van szó.

A merevlemezes alrendszer folyamatos működését különböző RAID megoldások szavatolhatják, de egy szerveralaplapi kártya meghibásodása a szerver leállását eredményezheti. Ha tehát van olyan szerver, amely képes a me-



— Érzem, hogy hamarosan törlődik az adatbázisom.

revlemezes alrendszert szoftveresen átvenni és az adatbázishoz a hozzáférést biztosítani, akkor ily módon folyamatos működést tudunk elérni. NT platformon ezt a Microsoft Cluster tudja megoldani. A rajta futtatható Oracle termék az Oracle Fail Safe (OFS), amely mind a Standard, mind az Enterprise adatbázisszerver változatnak része.

Az OPS NT platformon a Microsoft Cluster nélkül is hozzáférhető, és OPS konfigurációt workstation gépekből is kialakíthatunk. Hardver szempontból a Parallel Server és a Fail Safe között nincs lényeges különbség. Ahhoz, hogy a gépek egyidejűleg lássák a közös merevlemezes alrendszert, szükség van a Distributed Lock Manager (DLM) nevű szoftverre, amelyet általában a hardvergyártótól kell beszerezni. A DLM biztosítja azt, hogy az adatbázis konzisztens maradjon.



Az Oracle megoldásai

OPS alkalmazásakor a szerverek mindegyikén fut az adatbáziskezelő egy példánya, és a közös merevlemezes alrendszeren lévő adatokhoz egyidejűleg hozzáférhetünk. A felhasználó szempontjából mindegy, hogy melyik szerverhez kapcsolódik, ugyanazt fogja látni.

Az adatbáziskezelők saját memóriaterülettel rendelkeznek. A memória és a merevlemez közötti adatmozgatás legkisebb egysége a blokk. Mivel a memória adatelérési ideje nagyságrendekkel gyorsabb lehet, mint a merevlemez elérési ideje, ezért az adatbáziskezelő is igyekszik minél ritkábban fordulni a merevlemezhez. Íráskor és olvasáskor ezért egyszerre több adatblokkot olvasnak be a memóriába, illetve írnak ki a memóriából a merevlemezre. Olvasásnál azért, mert a rendszer arra számít, hogy nemcsak az éppen lekérdezett adatsorra lesz szükség az adatblokkból, hanem a későbbi olvasási igények érinteni fogják az adatblokk más sorait is, ha tehát azok már bent vannak a memóriában, nem kell újra a merevlemezhez fordulni. Írás esetében pedig a módosult adatblokkok összegyűjtésével és rendszeres időközönkénti tömbösített kiírásával lényegesen kevesebb időt vesz igénybe a merevlemezes kapcsolat.

Ha az adatbáziskezelő egyik példánya szeretne egy adatblokkhoz hozzáférni, akkor a DLM segítségével megkérdezi a többi példányt, hogy ott van-e valamelyiknél az adatblokk módosított változata. Erre azért van szükség, mert az Oracle adatbáziskezelő olvasáskor nem zárolja a sorokat, amiből a kérdés megválaszolható lenne. Ha nincs egyiknél sem, akkor beolvassa azt a merevlemezről. Ha valahol ott van, akkor két eset lehetséges. Ha ugyanazt a sort szeretné módosítani, amelyiken a másik dolgozik, akkor meg kell várnia, míg a tranzakció befejeződik vagy visszagörgetődik, és az adatblokk kiíródik a merevlemezre, ahonnan már be tudja azt olvasni. Ha az adatblokkon belül éppen nem használt másik adatsorról van szó, akkor a blokk azonnal kiíródik a merevlemezre. Ez az adatblokkátadási mechanizmus természetesen többlet erőforrást igényel.

További specialitások

Az Oracle8i rendszerben tovább finomították az algoritmust azzal, hogy ha csak olvasási célból van szükség az adatblokkra, akkor a hálózaton keresztül kapja meg azt az egyik példány a másiktól. Ha a rendszert sikerül az alkalmazás szempontjából úgy particionálni, hogy a különböző szerverekhez kapcsolódó felhasználók az adatoknak más-más körét használják, akkor nincs szükség adatblokkátadásokra.

Az Oracle 9i-ben nemcsak olvasás, hanem írás esetén is a hálózaton keresztül adják át az adatblokkokat. Ezzel közel lineárisan skálázható rendszereket lehet kiépíteni, vagyis ha a szerverek egyformák, akkor a fürt teljesítménye úgy is kiszámítható, hogy egy szerver teljesítményét megszorozzák a szerverek számával.

„Shared-nothing" esetén a teljesítmény növelésekor, vagyis egy új szerver beillesztésekor az adatbázist újra kell strukturálni. A „shared-disk" esetén erre általában nincs szükség. Képzeljünk el egy négysávos autópályát. Tegyük fel, hogy a forgalom növekedése indokolttá teszi egy újabb közlekedési sávval történő kiegészítést. A „shared-nothing" megoldáshoz az autópályát előbb le kell bontani, majd újjá kell építeni ötsávos változatban. A „shared-disk" modellben csak az a régi szélső sáv és az új szélső sáv közötti záróvonalat kell szaggatott vonalra átfesteni.

Ahhoz, hogy „shared-nothing" környezetben terhelésmegosztást tudjunk megvalósítani, megfelelően kell elosztani az adatokat az adatbázisok között, ami nem mindig egyszerű, ismernünk kell hozzá az adatok eloszlását. Egy üzletlánc esetében, ha minden hónaphoz egy gépet rendelnek hozzá, akkor a december havi forgalmi adatokat tároló gép terhelése többszöröse lehet a január hónaphoz rendelt gép terhelésének. „Shared-disk" esetén ez a probléma nem áll fenn, hiszen a két gép automatikusan megosztja a terhelést.

A „shared-nothing" megoldás önmagában nem tekinthető megbízhatónak. Bármelyik gép kiesése esetén az adatbázis általa kezelt része már nem érhető el. Ez „shared-disk" esetén fel sem merül, mert valamelyik gépen keresztül mindig elérhető a rendszer. A „shared-nothing"-hoz is hozzákapcsolhatunk hibatűrő megoldást (például Microsoft Clustert), de az átkapcsolási idő alatt ilyenkor is elérhetetlen a kiesett rész. A „shared-nothing" a „shared-disk" megoldással összehasonlítva skálázhatóság szempontjából sokkal kevesebb lehetőséget biztosít. Nem beszélve arról, hogy az utóbbi önmagában hordozza a megbízhatóságot.

Klotz Tamás
tamas.klotz@oracle.com

Az adatbázisok alatti mélyvíz

A teljesítémény hardverközeli forrásai

Ha valahol az adatbázisok kerülnek szóba, általában magáról az adatbázismotorról, annak szolgáltatásairól, hangolásáról, programozásáról vagy különféle felhasználási lehetőségeiről folyik az eszmecsere. Ezekben az esetekben az adatbázismotorok alatt elhelyezkedő bármiféle szoftvert és hardverkomponenst adottságként kezelünk. Csak nagyon ritkán és kevesen gondolják végig, mi is húzódik meg a relációs adatmotorok mögött. E cikkben elsősorban ezt a témakört, vagyis az operációs rendszereket, a számítógépek mikroarchitektúráját vagy éppen a processzorok tervezési és megvalósítási lehetőségeit vizsgálom, de mindvégig szem előtt tartom az adatbázisokat.

Az általános célú processzorok tervezését hosszú ideig egyetlen cél vezérelte: az új lapka legyen képes ütemciklusonként minél nagyobb számú utasítás végrehajtására. Mára ez a trend megváltozott, és elsősorban az alkalmazások leghatékonyabb futtatása a gyakorlati cél. A tervezési folyamatban ehhez természetesen szükséges mérni a jövőbeli processzor alkalmazási teljesítményét. A konstruktőrök azonban nem várhatnak a készülő prototípusokra, hiszen egy adott processzor esetében már a nagy léptékű alakzatok tervezési paramétereinek száma is eléri a több százat (ilyen például a funkcionális egységek száma, a gyorsítótár mérete és szervezése, vagy éppen a csővezeték lépcsőzése).

A paraméterek összessége együttesen és gyakorlatilag felbecsülhetetlen módon befolyásolja egy processzor teljesítményét. Ennek köszönhetően majdnem reménytelen, de mindenképpen költséges és időigényes eljárás lenne az összes szóba jöhető paraméterkombinációval megtervezni, ellenőrizni és legyártani egy-egy prototípust. A paraméterek kombinációjával elérhető maximális teljesítmény olcsóbb és mindenképpen gyorsabb becslésére ezért általában szoftverszimulátorok készülnek, amelyek ütemciklus-helyesen modellezik a processzorokat. Ezekben a szimulátorokban egyszerűen módosítható minden lényeges tervezési paraméter. A szimulátorokon azután a fejlesztők megpróbálják reprezentáns alkalmazások futtatásával megállapítani, hogy különböző szempontok szerint milyen paraméterkombináció tekinthető optimálisnak. Persze ezek a szimulátorok rendkívül lassúak, gyakran még a több száz elosztott processzoron futó rendszerek sem képesek elérni néhány Hz-nél magasabb egyenértékű órajelet, így csak másodpercenként 1-10 utasítás végrehajtását képesek elérni.

Processzorteszt

A processzorok tervezésénél leggyakrabban felhasznált mintaalkalmazások a SPEC teljesítménytesztek köréből kerülnek ki. Ezek a tesztek futásuk során általában milliárdos nagyságrendű számban hajtanak végre utasítást, vagyis egy teljes teszt lefuttatása évtizedeket venne igénybe. A megoldás a futtatandó alkalmazások mintavételezése, ún. nyomgenerálás. Ebben az esetben egy rendkívül egyszerű, nem órajelhelyes, de funkcióit tekintve korrekt másik processzorszimulátoron futtatják a programot. A funkcionális szimulátor általában MHz-es nagyságrendű, egyenértékű teljesítményű, és fájlba kiírja a teljes program futására vonatkozó adatokat. A teljes futási adatbázist utána egy feldolgozó program megemészti, kialakítja a néhány millió utasításból álló, statisztikailag korrekt mintát, és az már lefuttatható a ciklushelyes szimulátoron.

A mintavételezett nyom (trace) ad lehetőségét arra, hogy a napjainkban tervezés alatt álló processzorok esetében a tervezők már ne csupán a SPEC és a hasonlóan egyszerű többi teljesítméndefault teszt mintáit vegyék figyelembe, hanem kíséreljék meg a tipikus adatbázisfeladatok nyomainak futtatását is. Az adatbázisnyomok első eredményei mára már publikálásra kerültek. Fontos azonban megjegyezni, hogy ezek sajnos a processzorok mikroarchitektúráján túl több felsőbb réteg tulajdonságait is magukon viselik. A nyomok futásának eredményeként ma a 8–16 MB-os másodszintű, de nem integrált, vagy az 1–2 MB-os másodszintű integrált gyorsítótárak tűnnek megfelelőnek, de ez is a technológia által megszabott korlát.

Címtranszformáció

Érdekes eredmények mutatkoztak meg a hardveres címtranszformációt elvégző TLB-k tekintetében is. A TLB-k magasabb fokú asszociativitása mindenképpen előnyös az adatbázisalkalmazásokhoz, a TLB bejegyzéseinek számát tekintve pedig az 1024 feletti teljesítménynövelő hatása jelentős. Ennek indoka igen egyszerű: az adatbázisok általában nagy mennyiségű adatot tartanak a memóriában bizonyos műveletek elvégzéséhez. A sok memória több lapon helyezkedik el, más-más fizikai memóriacímek tartoznak hozzá, és ha a processzor nem képes ezeket saját TLB struktúrájában tárolni, akkor a memóriában tárolt további táblákhoz kell fordulnia.

Egy másik terület, ahol az adatbázisok hatása jól érzékelhető: az elágazás-előrejelző (predikciós) logika. Az adatbáziskód — csakúgy mint az egész számokkal numerikus műveleteket végző többi program — általában nagy számú elágazó utasítást tartalmaz, ezért a megbízhatóbb ugrásbecslés elengedhetetlen. Az adatbázisoknak azonban speciális tulajdonsága a halmozott ugrás, ahol egy elágazási cél utasítása egy újabb feltételen nyugvó következő elágazás, majd ennek célja egy harmadik, és annak esetleg még egy negyedik is. Ez a működési szabályszerűség számos optimalizálási lehetőséget rejt magában, ami azonban még sok további kutatást igényel.

Az adatbázisok processzorszintű működésének megértésében további segítséget nyújtanak a mára már szinte minden nagyteljesítményű processzorban megtalálható számlálók. Ezek valójá-

ban speciális regiszterek, amelyek képesek előre meghatározott 10–25 esemény bekövetkezését számlálni (gyártótól függően). Ilyen események lehetnek az eltelt órajelciklusok, a végrehajtott utasítások, a gyorsítótár találatok vagy a nem találatok száma és egyéb paraméterek is. A teljesítményszámlálók felhasználásának egyik legátgondoltabb és legkönnyebb, magas szintű nyelvből (például C) forráskódból és parancssorból is elérhető megvalósítása a Sun Solaris 8 operációs rendszerében található. Ezek a teljesítményszámlálók lehetőséget adnak az adatbázisok valósidejű futási viselkedésének megfigyelésére.

Összeköttetések

A hardver következő rétege a többprocesszoros rendszerek esetében válik érdekessé. Ennek kapcsán nemcsak a multiprocesszoros rendszerek gyorsítótárának konzisztenciáját biztosító megoldásokról, ahanem általában az összeköttetési technológiáról is szót kell ejteni.

A modern összeköttetések egyik jellemzője a csomagkapcsolt technológia, amelynek lényege, hogy az összeköttetési hálózat (legyen az akármilyen fizikai topológiájú) csak a kérések vagy az azokra adott válaszok továbbításának idejére válik foglalttá. Mindez a multiprocesszoros rendszerek esetében igazán érdekes, ebben az esetben ugyanis a párhuzamosan dolgozó processzorok nem foglalják le egy adattranszfer idejére a teljes összeköttetési hálózatot, tehát nem sorozatosítják az egyébként párhuzamos végrehajtást. Az adatbázismotorok feladatukból adódóan képesek a párhuzamos működésre, így tudnak, sőt „szeretnek" multiprocesszoros rendszereken futni. A csomagkapcsolt rendszerek ezért itt óriási előnyt élveznek. A csomagkapcsolás mellett a kapcsolat topológiája is érdekes teljesítménynövelő technológiák alkalmazására ad lehetőséget.

Az általános célú modern multiprocesszoros rendszereknél általában a busz és a pont-pont kapcsolat keverékéből kialakuló hibrid crossbar rendszer a leginkább célravezető. Az ilyen hibrid topológia (például a Sun Enterprise 10000 szerver és az UltraSPARC-III processzorra épülő SunFire sorozat) általában busz jellegű megvalósítást alkalmaz a memóriacímek multicast elosztására, ami azért szükséges, mert a konzisztencia biztosításához szükséges módosítások elvégzéséhez minden processzorhoz tartozó gyorsítótár-vezérlőnek értesülnie kell a memóriatranzakciók címéről (ez a lényege a snoop, vagy magyarul hallgatózás alapú konzisztenciaprotokolloknak).

A konzisztenciaprotokollok minimális megoldásaként egy 4 állapotú diszkrét automatát szokás alkalmazni, amelynek neve az állapotok kezdőbetűjéből alkotott mozaikszó, a MESI. A minimális állapotokon túl minden további állapot valami optimalizálásra ad lehetőséget, és noha ezek nem minden esetben köthetők adatbázis jellegű felhasználáshoz, számos adatbázisfeladat esetében teljesítménynövekedést jelentenek. A leggyakoribb kiegészítés az 5 állapotú MOESI protokoll, de ismeretesek 7 és 9 állapotú speciális protokollok is. Sajnos az állapotok számának növekedése együttjárhat a tranzakciók bonyolultabbá válásával, így paradox módon a teljesítmények romlásával is. A címbusz és általában a busz topológia komoly korlátokkal jár. Ezek a korlátok leginkább a buszra kapcsolódó hallgatók (azaz az alaplapok és a processzorok) számával, valamint a busz hosszával függenek össze, így nem skálázhatók a végtelenségig. A technológia mai határa 64 processzornál húzódik.

UMA, NUMA

Az adatkapcsolatok esetében a pont-pont kapcsolatok kiépítése a célravezető, hiszen ilyenkor általában nincs szükség az adatok multicastolására (valójában ez a konzisztenciaprotokoll kérdése). A pont-pont kapcsolatok jelentősen tudják növelni a rendszerben a memória sávszélességét, ami viszont az adatbázismotorok párhuzamos működéséből és a felhasználás jellegéből adódó nagy memóriafogyasztás miatt igen hasznos tulajdonság. Az ilyen hibrid megoldások óriási vonzereje, hogy a memóriahozzáférés a teljes rendszerben a kérdező és a válaszoló fizikai helyétől függetlenül konstans, és csak a tranzakció típusától függ. Ezeket a rendszereket szokás UMA (Uniform Memory Access), azaz egységes memóriahozzáférési idejű rendszereknek nevezni. Az UMA rendszerekben ezen tulajdonságnak köszönhetően sem az operációs rendszereknek, sem az alkalmazói programoknak nem kell tudniuk a fizikai memória elhelyezkedéséről, hiszen az nem érinti a teljesítményt.

Ezzel ellentétben egy NUMA (vagyis nem egységes memóriahozzáférési idejű) rendszer esetében valakinek gondoskodnia kell arról, hogy a memória allokációja optimálisan megtörténjen.



— Mikor frissítették utoljára az adatbázisukat?

Ez sajnos rendkívül bonyolult feladat, hiszen a programok futása során a memória felhasználási mintázata dinamikusan változik, különösen egy adatbázisszerverben, ahol gyakorlatilag minden lekérdezés egyedi. A NUMA rendszerek ennek megfelelően általában ott érnek el szép teljesítményt, ahol előre kitalálható a memória optimális elosztása, vagyis az adatok lokalitása ismeretes. Ilyen alkalmazások lehetnek egyébként a numerikusan intenzív feladatok (például a mátrixműveletek).

A NUMA rendszerek előnye, hogy skálázódásuknak tulajdonképpen nincs határa. Ennek megfelelően a NUMA rendszerek konzisztenciájának megvalósítása sem a korábban említett hallgatózás alapú, hanem könyvtár (directory) jellegű. A directory alapú protokollok legnagyobb hibája viszont, hogy ha kifutnak a rendelkezésre álló memóriából, akkor rendkívül lelassulnak. Ez különösen a véletlenszerű mintázatban és hozzáférésben nagy memóriát lefoglaló alkalmazások, vagyis az adatbázisok esetében kellemetlen. A modern összeköttetések másik alapvető újdonsága az elsőként a Cray SuperServer 6400 rendszerben megjelent particionálási lehetőség, amely napjainkban a Sun Enterprise 10000 szerverében él tovább.

A Sun koncepciója

A particionálás alapja, hogy az összeköttetési hálózat képes csak a fizikai memória címe alapján meghatározott alaplapoknak továbbítani az adatokat és a tranzakciókat. Ennek eredménye, hogy a számítógép buszán hallgatózó alaplapok egy particionált rendszerben kizárólag a saját particiójukhoz tartozó memóriacímekre és adattranszferre figyelnek. A particionálás így tökéletes izolációt biztosít, vagyis a számítógép virtuálisan több önálló, izolált rendszerre bontható. A Sun jelenlegi és a Cray korábbi megoldásában a partíciók dinamikusan változhatnak, új partíciók alakíthatók ki egy működő rendszerben, a meglévők átméretezhetők vagy meg is szüntethetők. A Sun Enterprise 10000 további előnye, hogy a fizikai memória alapján történő útvonalválasztás eredményeképpen képes olyan memóriarégiók kialakítására, amelyek egyszerre több partícióhoz is tartoznak. Egy ilyen közös memóriaterület kommunikációs közegként működhet két partíció között. Ennek a megoldásnak a felhasználásával GB/sec-os átviteli csatornákat lehet kialakítani két virtuális rendszer között. A partíciók kialakíthatósága nemcsak az adatbázisrendszerek konszolidációjában, hanem működésük optimalizálásában is rendkívül hatékony eszköz lehet.

A Sun particionálási koncepciója az UltraSPARC-III processzor alapú SunFire termékcsaládban már a legkisebb vállalati szerverektől kezdődően megjelenik, kiindul a 8 processzoros rendszertől és egészen a legnagyobb, 72 processzoros szuperszerverig terjed. A particionálási technológia végül kilép a számítógépből, és megjelenik a számítógépeket összekötő hálózati technológia részeként. A Sun fejlesztéseinek egyik eredménye egy rendkívül nagy sebességű optikai kapcsolat. Ez lehetővé teszi, hogy akár több ezer processzorból és nagyon sok számítógépből álló környezetetet kiszolgáljanak egyetlen operációs rendszer futtatásával. Másrészt ez a technológia az adatátvitelen túlmenően alkalmas az erőforrások particionálására is.

A fenti új technológia kapcsán a memóriahozzáférés rendszerét újra érintenünk kell. A NUMA rendszerek adatbázisoknál fellépő problémáit már ismertettük, és láttuk a UMA skálázódási korlátait is. Tehát valami mást kell alkalmazni. A NUMA és az UMA technológia előnyös tulajdonságait egyesítő architektúra adja a megoldást: a COMA (Cache Only Memory Architecture), vagyis a csak gyorsítótár alapú memória. A COMA esetében a teljes fizikai memóriát a gyorsítótár elvei szerint kezeljük, a memóriát sorokba szervezzük, és minden sorra nyilvántartjuk annak konzisztenciáját. A COMA előnye, hogy lehetővé teszi adataink többpéldányos meglétét, így rövid idő elteltével nagyon jó adatlokalitást érhetünk el. A COMA rendszerek megvalósítása alapjaiban hardverrel vagy szoftverrel támogatott megoldás lehet, ez utóbbit nevezzük egyszerű vagy sCOMA rendszernek. A COMA architektúra felhasználásával lehetőség nyílik lényegesen nagyobb adatbázisok és feladatok hatékony, de egyszerű megoldására, hiszen nem kell adatbázisunkat felkészíteni az eltérő memóriahozzáférési viszonyokra.

Fürtök és fordítók

Az operációs rendszerek számos eszközzel tudják javítani vagy rontani adatbázisaink teljesítményét. A legtöbb operációs rendszer alkalmaz a fájlrendszerre gyorsítótárat, és az adatbázisok is hasonlóképpen dolgoznak. A két gyorsítótár felesleges memóriafoglalást, a teljesítményben pedig problémákat tud okozni, ezért bizonyos operációs rendszerek támogatják a direkt (a fájlrendszer gyorsítótár funkcióit kikerülő) I/O műveleteket is. Persze ismeretes megoldás az úgynevezett nyers eszközökre (raw device) telepített adatbázis, amely egyértelműen kikerüli a fájlrendszerek gyorsítótárait. Hasonlóan a fájlrendszer szintjén mozgó, egyszerű, de hatékony trükk az aszinkon I/O. Ebben az esetben az operációs rendszer később írja ki lemezre a blokkot, így nem iktat be várakozási időt egy amúgy is zsúfolt pillanatban. A Unix rendszerek további jelentős optimalizálási lehetősége magának a gyorsítótáras fájlrendszernek a kialakítása, ezáltal speciális allokációs mechanizmusokat lehet alkalmazni abban a pillanatban, amikor hirtelen nagy memóriaigény lép fel egy alkalmazás részéről.

Hasonló optimalizálási lehetőség, ha az operácios rendszer képes együttműködni az adatbázissal, és működés közben meg tudja változtatni a megosztott (shared) memóriaterületek méretét. Ez a dinamikus memóriaarchitektúra az adatbázisok optimalizálása érdekében történt egyik legújabb fejlesztés, és különösen a dinamikusan rekonfigurálható és bővíthető rendszerekben nagy jelentőségű.

Az sCOMA architektúra esetében az operációs rendszer speciális megoldásai biztosíthatják az adatok replikációját. A COMA rendszerekben általában megvalósítható a processzek ütemezésének affinitás alapú ütemezése. Az affinitásütemező igyekszik megbízható információt szerezni és tárolni a processzek legutolsó futási helyéről. Vagyis arról, hogy egy processz melyik memóriatartományhoz képest futott lokális processzoron, hiszen a memóriareplikátor valószínűleg már átmozgatta a szükséges adatokat. Később, ha egy processz újra futtatható állapotba kerül, az affinitásütemező a lehetőséghez képest igyekszik megoldani, hogy a processz újfent a korábbi lokális processzoron fusson, így kihasználhassa a már megszerzett lokalitást.

A nagy adatbázisok számára jó megoldás a párhuzamos szerverek építése is, mert számos lehetőség alakítható ki a teljesítmény növelésére — kezdve a fürtszintű fájlrendszerektől egészen a parallel szerverek gyorsítótárainak fúziós megoldásáig. Végezetül fontos faktora lehet a teljesítmény növelésének a legfrissebb optimalizáló fordítók használata, bár ezen a téren a gyártók egyelőre inkább a konzervatív megoldást részesítik előnyben.

Fischer Erik
erik.fischer@hungary.sun.com

Egy mindentudó család

IBM DB2 Universal DataBase

Az IBM DB2 Universal Database több évtizede jelen van az adatbáziskezelők piacán. Nagygépes környezetből indult, de ma már minden elterjedt platformot támogat, széles skálán mozog, és a legújabb technológiák követelményeinek is megfelel.

Az egyes rendszerkörnyezetek és felhasználási területek eltérő igényeit a DB2 különböző változatai (edition) elégítik ki.

A DB2 változatai

— Ha a kisebb gépektől haladunk az egyre robusztusabb megoldások felé, akkor első helyen említhető a **DB2 Everywhere**, amely PDA marokszámítógépek (personal digital assistant) és HPC kéziszámítógépek (handheld personal computer) adatkezelését oldja meg. Itt természetesen csökkentett funkcionalitás érvényesül. Támogatott operációs rendszerek: a Palm és a Windows CE.

— A **Satellite Edition** 32 bites Windows operációs rendszerekhez készült, olyan gépekhez, amelyek nem csatlakoznak folyamatosan valamilyen központi adatbázishoz. Például az ügynöki munkaállomások tipikus adatbáziskezelője lehet.

— A **Personal Edition** a hálózati kiszolgálás nélküli megoldások alapja, általában egyedi speciális megoldásoknál használatos.

— A **Workgroup Edition** kis felhasználói csoportok adatkezelését oldhatja meg, ahol egyidejűleg viszonylag kevés kiszolgálást kell végezni, illetve az adatbázis mérete sem igényel nagy erőforrásokat. A támogatott operációs rendszerek közül a Windows NT és 2000 mellett a Linux, az AIX, az OS/2, a HP-UX és a Solaris platformot érdemes kiemelni.

— Az **Enterprise Edition** alkalmazható nagyobb rendszerek esetén, és ahol egyidejűleg több száz kiszolgálási igény érkezik az adatbázishoz, illetve az adatok mennyisége is nagyobb. Tipikus platformja a Unix (AIX, HP-UX, Solaris, Linux stb.), de természetesen ez a változat is megvan Windows operációs rendszerre.

— A legnagyobb adatbázisokhoz az **Enterprise Extended Edition** a megfelelő, amely ki tudja használni a nagy rendszerek architektúrájának előnyeit (MPP feldolgozás) és segítségével megvalósíthatók az osztott adatbázisok, a párhuzamos lekérdezések is.

— Külön kategóriaként lehet figyelembe venni magát az **AS/400** és **S/390** környezetet, melyekben a DB2 természetesen szintén jelen van.

A DB2 jelenleg a 7.1-es verziójánál tart, és az számos újdonságot tartalmaz az előző változatokhoz képest. A mai körülmények között már nemcsak azt várják el egy adatbáziskezelőtől, hogy megbízhatóságú, jól skálázható és könnyen kezelhező legyen. Emellett támogatnia kell a legmodernebb technológiákat, tartalmaznia kell számos kiterjesztést, kiegészítést, hogy hatékonyan használhatók legyenek. Nézzünk néhány példát ezekre a továbbfejlesztésekre.

E-business

Az e-business alkalmazások dinamikusan fejlődnek és tért hódítanak. Az adatbázisoknak is támogatniuk, követniük kell ezt a tendenciát. A DB2 számos olyan új jellemzővel rendelkezik, melyek segítik ezen alkalmazások fejlesztését, üzemeltetését.

XML-támogatás

Lehetőség van XML (eXtended Markup Language) adattípus létrehozására, így egyszerűen tárolhatunk például weboldalakat. A gyorsabb keresés érdekében indexeket építhetünk a különböző dokumentum elemekre és attribútumokra.

Net.Data kiterjesztés

A Net.Data, a webkiszolgálók és a DB2 összeköttetését megvalósító alkalmazás mostantól beépített XML támogatást tartalmaz. Ezzel hatékonyan hozhatók létre kimeneti stíluslapok (XSL) a megjelenítés formájának meghatározására.

Tárolt eljárások

A DB2 esetében nincs megkötve a tárolt eljárások nyelve, lehet az C, C++, Java stb. A tárolt eljárásokat készítő eszköz (stored procedure builder) számos új funkcióval bővült. Lehetőség van beágyazott tárolt eljárások használatára is, amelyekkel a fejlesztés sok



— A fiúk az edatbányában dolgoznak...

esetben leegyszerűsíthető. Az SQL eljárásnyelv (procedure language) mostantól nemcsak mainframe és AS/400 környezetekben használható, hanem minden támogatott platformon. Az alkalmazásfejlesztési kliens (application development client) tartalmaz számos tárolt eljárásmintát különböző programozási nyelveken. A Java osztálybetöltő (Java stored procedure class loader) segítségével szintén növelhetjük a hatékonyságot, hiszen lehetőség van az egyes osztályok működés közbeni cseréjére.

Üzleti intelligencia

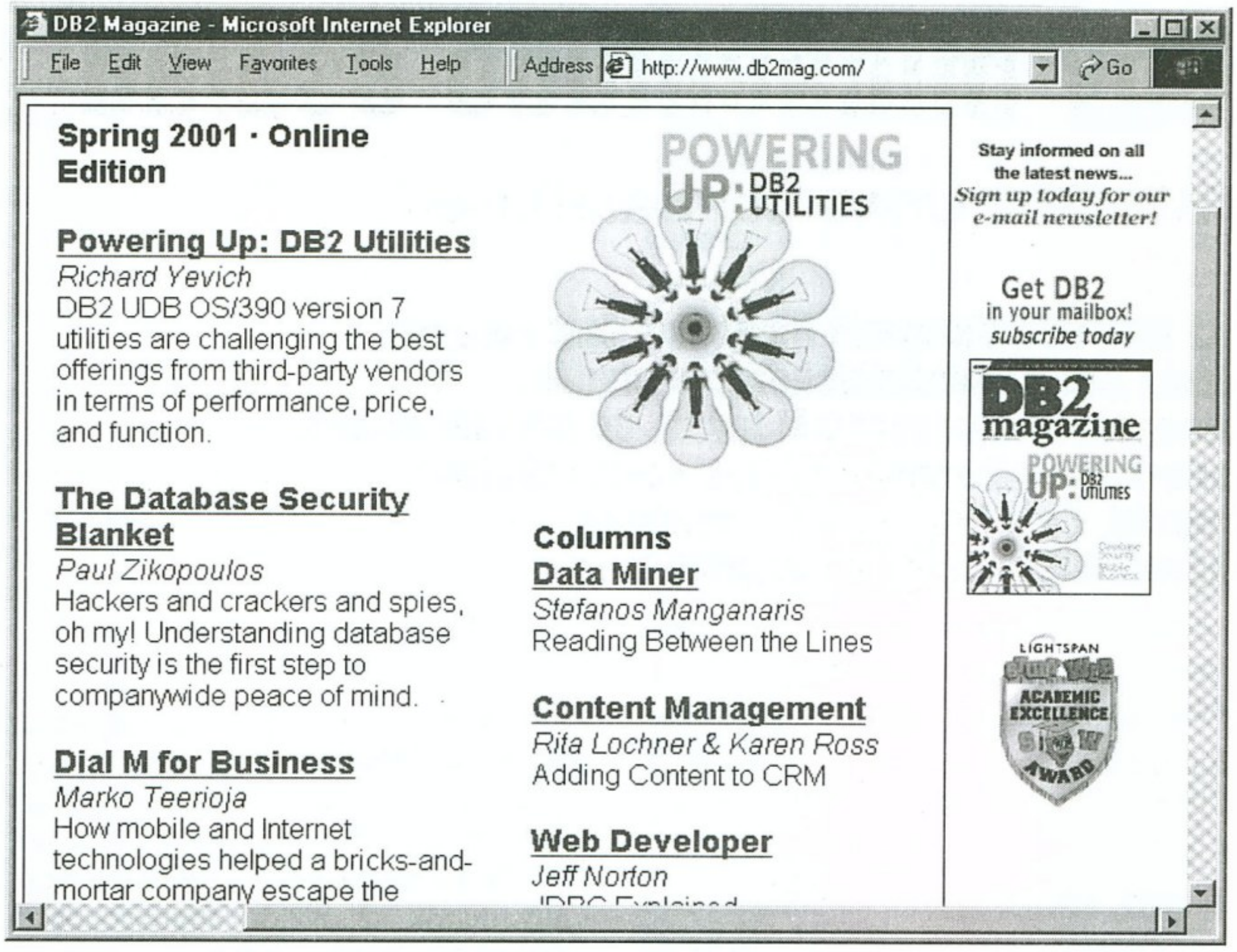
Napjainkban minden területen érezzük, hogy az üzleti intelligencia által nyújtott lehetőségeket a hatékonyabb működés érdekében ki kell használnunk. Üzleti adatainkból ki kell nyerni a lehető legtöbb értékes információt, és ezek ismeretében kell meghoznunk döntéseinket. A DB2 számos eszközzel támogatja az üzleti intelligencia jellegű felhasználásokat.

— A Data Warehouse Center a 7-es verziótól része a DB2-nek. Segítségével elérhetők a különböző adatforrások, definiálhatók az adatmozgató és adatátalakító folyamatok (több száz transzformáció), és kezelhetők segítségével a metaadatok is. Támogatja a teljes és a növekményes adatfrissítést, ezáltal az adattárház adminisztrációs feladatai lényegesen egyszerűbbé válnak.

— Az OLAP Starter Kit a többdimenziós integrált adatelemzés megvalósítására szolgál. Tartalmazza a Hyperion Essbase technológián alapuló DB2 OLAP server funkcionalitásait, és csak az egyidejű felhasználók száma korlátozott. Segítségével könnyedén készíthetünk multidimenzionális alkalmazásokat, és hozzájuk szükséges adatszerkezetet. Relációs OLAP adatbázis kialakítására is van lehetőség, amellyel SQL függvények is használhatók a számításokhoz. Ilyenkor csillag séma szerint tároljuk az adatokat a relációs táblákban.

— A QMF (Query Management Facility) segítségével könnyedén elkészíthetők különböző lekérdezések, riportok. Ezek eredménye integrálható OLE2 eszközökkel, például táblázatkezelőkkel, diagram- és elemzéskészítőkkel, stb.

— A Query Patroller felügyelheti, kezelheti, ütemezheti az összes dinamikus SQL lekérdezést. Használatával nagy adatbázisrendszerek esetén is központi helyről vezérelhetők az összetett műveletek. A klinesoldali kódba épített nyomkövető pontok segítségével a kiszolgálón megfigyelhető az SQL végrehajtás.



Adatkezelési kiegészítések

Számos olyan újdonsággal találkozhatunk a DB2 legújabb verziójában, amelyek általános jellegűek, és minden felhasználási területen alkalmazhatók.

Átalakítás statikus SQL-lé

Az ODBC lekérdezések statikus SQL parancsokká alakíthatók. Ezzel akkor érhetünk el jobb teljesítményt, ha ugyanazt az ODBC lekérdezést gyakran futtatjuk.

Objektumrelációs jellemzők

Kibővült a stukturált adattípusok támogatása, így a táblák már strukturált típusú oszlopokat is tartalmazhatnak. Lehetőség van strukturált adattípusok egymásba ágyazására is, mivel az adattípus attribútumok nem korlátozódnak az SQL alaptípusokra, így az egyes típusok egymásból származtathatók.

A felhasználó által definiált indextípusok

Ezek segítségével saját módszer alakítható ki az indexek karbantartására, keresésére és kiaknázására. Lehetőség van például adatbázisban tárolt térbeli és egyéb összetett adatokhoz készíteni indexet.

Adatkapcsolatok

Az adatkapcsolat egy adatokat tartalmazó fájlra mutató csatolás. Ezekben a fájlokban gyakran multimédiás tartalom van, így igen nagy méretűek is lehetnek, és a módosítások igen nagy hálózati forgalmat generálhatnak. A Data Propagator termékkel lehetőség van a frissítések konfigurálására, és így a szükséges hálózati forgalom optimalizálható.

Natív OLE DB támogatás

Ennek segítségével elérhetők az OLE DB adatbázisok, valamint ilyen kérések kiszolgálására is képessé tehető a DB2. Táblafüggvények segítségével adatokat tölthetünk a DB2-be, valamint könnyen megvalósíthatjuk ezek elérhetőségét.

SQL Assist

Grafikus felületen keresztül, minimális SQL nyelvi ismeretekkel is összeállíthatjuk összetett utasításainkat, legyenek azok akár lekérdezések, módosítások, adatbeviteli vagy törlési utasítások.

Megnövelt naplófájl

A naplófájl maximális mérete 32 GB-ra nőtt. Ez már az összetett, hatalmas adatmennyiséget érintő tranzakciók létrehozásában lehet segítségünkre.

Térinformatika

A DB2 Spatial Extender segítségével térinformatikai alkalmazások alapja is lehet a DB2: térbeli tulajdonságoknak megfelelő funkciók és adattípusok hozhatók létre. Támogatja mind az OpenGIS, mind az ISO tárolási, indexelési és lekérdezési szabványokat.

Kovács László
laszlo.kovacs@hu.ibm.com

Stabilitás a változó környezetben

A DataFlex fejlesztési filozófiája

A fejlesztőrendszerektől joggal elvárható, hogy produktumaik nagyobb befektetés nélkül illeszkedjenek a folyamatosan változó környezethez. Az új technikai lehetőségeket minden alkalmazás számára hasznosítani kell, ezért fontos olyan eszközt választani az adatbáziskezelő alkalmazás fejlesztéséhez, amely megkönnyíti az átjárást a platformok között, és egy-egy „rendszerváltáskor" vagy más adatbázisformátumra való átállás esetén sem szükséges a nagyobb átdolgozás. Ilyen eszköz a DataFlex.

A Data Access Corporation által 1976 óta folyamatosan fejlesztett DataFlex programcsalád erőssége, hogy több operációs rendszeren egyaránt „meg tud élni". A DOS és a Console Mode verzió lefedi a hagyományos DOS-os (MS-DOS 6.2x) és a Windowsban is meglévő karakteres alkalmazási igényeket (Windows 98 SE, NT 4 és 2000). Ebben a felhasználói körben elsősorban a nagy tömegű adatbevitel biztonságos formáját keresik. A szükséges előkészítő műveleteket persze el kell végezni, hogy rögzítés közben ne legyen fennakadás, ne kelljen folyton korrigálni.

A Unix és Linux rendszerre készített DataFlex verziók elsődlegesen szintén a karakteres alkalmazások iránti igényt elégítik ki. Három formában jelenik meg: RISC processzoros gépekre (IBM AIX 4.2 és magasabb verzió, Sun Solaris 2.x), Intel processzoros gépekre (SCO Unix 3.2 v4.0 és magasabb verzió, UnixWare 7.0.1 és magasabb verzió) és Linuxra (Linux systems with Kernel 2.2.14 és magasabb verzió).

A Visual DataFlex a windowsos világban teszi lehetővé az objektumok elérhetőségét anélkül, hogy azokat különösebben ismerni kellene. Ehhez megfelelő támogatást adnak a Data Access által készített felhasználóbarát objektumosztályok, amelyek lefedik az adatbevitellel kapcsolatos feladatokat, az adatbázis alkalmazásban megjelenő listák, grafikonok, kimutatások, statisztikai táblák kialakítását.

Az internet világában, a webes alkalmazásokban a dinamikus HTML kezeléséhez két teljesen különböző filozófiájú felület funkcióit kell egységes formában kezelni. Ehhez a WebApp Studio segítségével lehet például adatbáziskezelő eljárásokat készíteni, és a kétféle felület közötti átjárást a WebApp Server biztosítja — jelenleg Windows NT környezetben, de hamarosan Linuxon is.

A gyártó által rendelkezésünkre bocsátott objektumosztály-halmaz az általános szempontokon túl sok különleges igényt is kielégít. Az egyszerű adatbeviteli ablakoktól kezdve a táblázatok kezelésén át a számlaformátumú bizonylatrögzítésig a legkülönbözőbb formákat lehet kialakítani (jelentéskészítés, grafikus ábrázolás, OLE objektumok stb.). Az objektumosztályok tetszés szerint bővíthetők az objektumokkal szemben támasztott tartalmi követelményeknek megfelelően (polimorfizmus, encapsulation stb.).

A Visual DataFlex az XML állományok kezelésével lehetővé teszi a platformok közötti átjárást, és teljesen platformfüggetlen felületen keresztül kommunikál. A létrehozott alkalmazásokban biztosítható az internetes elérés (web, levelezés, ftp stb.), továbbá a helyi hálózaton lévő és a távoli gépek megcímzése is.

Az adatbázisban tárolt adatok visszanyerését, érdemi outputok készítését a beépített jelentéskészítők segítik. Karakteres módban egyszerű és összetett listák készítéséhez a DFQUERY ad támogatást, Visual DataFlexben pedig a Seagate Software által fejlesztett Crystal Reports 8 teszi lehetővé a látványos eredménytáblák elkészítését. (Az adatbázisban előre nem definiált összefüggések szerint is.)

Amikor valahol operációs rendszert vagy adatbázisformát váltanak, gazdaságossági számításokat is szoktak végezni. Sok tervet meghiúsíthat, ha az alkalmazások átállítása túl nagy költséggel jár, mert a fejlesztőket hosszas, bonyolult (vagyis költséges) munkára kényszeríti. A karakterorientált felület-



— Az adatbázis helyett inkább az adatrögzítőt tartanám karban...

ről grafikus környezetre való áttérés elkerülhetetlenül többletmunkával jár. Az viszont megkérdőjelezhető, hogy miért kell az adatbáziskezelést is átírni, ha annak nincs semmi különleges indoka. Az adatbázisok kezelésének szabályai egységes elveken alapulnak (rekordok készítése, módosítása és törlése, a reláció alacsonyabb szintjén elhelyezkedő tétel módosítása által előidézett változás magasabb szinten, a mezők ellenőrzése, a hibaüzenetek és a segédlisták kezelése stb.). Az azonos szintaktika alapján egy alkalmazás áttétele egyik operációs rendszerről a másikra nem igényelhet nagyobb erőfeszítést. A DataFlex olyan modulok, csomagok készítését teszi lehetővé, amelyek „egy az egyben" használhatók a különböző platformokon.

A Unix-verziók közötti átjárás általában csak a programok újrafordítását igényli, feltételezve, hogy „előéletükben" nem kezeltek Unix-környezettől függő speciális programrészeket. DOS-ról Unixra hasonlóan egyszerű az átmenet, de itt a karakterkészlet kezelésének beállításával a paraméterezés szintjén valószínűleg külön is kell foglalkozni. (A programmal általában viszont nincs semmi teendő.)

A DataFlex saját adatbázismotora a kisebb és a nagyobb adatbázisokat is önálóan, hatékonyan működteti. Kérdés azonban, hogy mi a helyzet, ha a felhasználó csatlakozni akar olyan rendszerhez, amely más adatbázisformát használ, vagy ha kliens-szerver architektúrát akar kialakítani. A DataFlexhez kapható összekötő készletekkel (connectivity kit) a probléma elegáns módon, különösebb programozási munka nélkül megoldható. Visual DataFlex és WebApp Server esetén Windows Console Mode-ban ilyen kész csomagokkal elérhető az Oracle, a Pervasive SQL, az MS-SQL, az IBM DB2 Universal Database és az ODBC. Unix alatt az IBM DB2 Universal Database-hez lehet kapcsolódni. Az összekötő készletek és a különböző segédeszközök (utility) kezelik a fájldefiníciós funkciókat, megoldják az adatbázis konvertálását, és lebonyolítják az újraszervezési eljárásokat.

A felhasználói programok szintjén semmit nem kell módosítani. A fejlesztéskor nem is foglalkoznak azzal, hogy milyen adatbázisformát használnak majd a gyakorlatban. A kész programokhoz csak a megfelelő driverekről kell gondoskodni, és azonnal alkalmazható az új adatbázisforma.

Michaletzky Géza
nextsw@nextsw.hu

Vállalati rendszerek

Az Ablesoft koncepciója

Néhány fős cégektől kezdve a többezres mammutvállalatokig az adatok kezelése, tárolása, védelme és különböző szempontok szerinti lekérdezése alkotja a működés gerincét. A gazdasági vezetőknek mindig a konkrét információkra támaszkodva kell tervezniük és dönteniük. Az alábbiakban a Simpletech vállalatirányítási rendszerén, az Ablesoft program példáján keresztül mutatjuk be ennek néhány összefüggését.

Mivel a vállalatok alapvetően a rendelkezésükre álló információk előre meghatározott körével és típusaival dolgoznak, szükséges azokat egységes rendszerben feldolgozni. A rendszerezés kezdetén a működési hasonlóságok és a törvényi előírások jelentik a kiindulópontot, később a konkrét sajátosságok ismeretében az információs rendszerek fejlesztői részletesebb lekérdezési módokat építenek be, hogy minél teljesebb legyen a vállalatirányitási szoftver által nyújtott szolgáltatás.

A főbb adattípusokat általában minden cég használja. A szoftver „lelkét" jelentő termelésirányitási mag köré a készlet- és raktárgazdálkodási, az ügyviteli és a személyügyi modul illeszkedik. Az egyes modulokban fellelhető adatok az adott területekről szolgáltatnak információt, de egy közös adatbázisban helyezkednek el, ezért nem szükséges a modulok között külön kapcsolatokat létesíteni ahhoz, hogy az információk felhasználhatók legyenek más modulokban is. Az integrált adatbáziskezelők éppen azt az igényt elégítik ki, hogy szükség esetén az összes információ lekérdezhető és feldogozható legyen.

A hatékony döntéshozatal érdekében a vállalati adatbázisrendszer az adatokat összefüggő adathalmazként kezeli, és a módosításokat láncreakciószerűen végiggörgeti a teljes rendszeren, a változásban érintett összes adaton. Így néhány gombnyomással bármikor képet kaphatunk a pillanatnyi helyzetről, válogathatunk az információkból aszerint, hogy milyen üsszefüggésekre, milyen képletek, egyenletek, elemzések eredményére vagyunk kíváncsiak.

A rendelkezésre álló adatokat az adatbásikezelő rendszer olyan táblázatban tárolja, amelynek oszlopai az adatokat logikai, tartalmi eltéréseik alapján, sorai pedig fizikai, nagyságbeli különbözőségük szerint rendezi. Ezen belül a rendszer az adatokat további utasításokkal alapadatokra és forgalmi adatokra bontja, amire azért van szükség, hogy elkülönüljenek a stabil alapadatok (például a terméknevek) a folyamatosan változó forgalmi adatoktól (amilyenek például a számlákban vannak). A rendszer ennek megfelelően különválasztja az alapadatok kezelésének funkcióit a forgalmi adatokat kezelő funkcióktól. Az alapadatok elsősorban a bevezetés szakaszában játszanak nagy szerepet, a forgalmi adatok inkább a rendszer folyamatos működése során.

Ahhoz, hogy megbecsüljük egy adatbáziskezelő rendszer hardverigényét, kellőképpen tájékozottak kell lennünk a benne szereplő adatok nagyságáról. Az adattípusonkénti logikai elkülönítéshez az Ablesoft rendszerben 300 forma található. Ezek közül a legkevesebb információt rendszerint a postai irányítószám tartalmazza, az csak a földrajzi elhelyezkedésre utal. A legnagyobb — például egy termelő vállalat esetében — általában a munkaszámforgalmi állomány. Az utóbbi komplex táblázat a gyártott és forgalmazott termékről nyújt sokrétű információt. Oszlopainak száma átlagosan eléri a 100-at, a sorok száma pedig a rendszer használatával egyre csak nő, egy év alatt a 3000-et is meghaladhatja. Az oszlopok határozzák meg a rendelt termék cikkszámát, a rendelt mennyiséget, a határidőt, a technológiát, az MRP módját vagy az ütemezést stb. Magán az MRP-n belül is számos részinformáció található: a gyártás anyagszükségletének tervezése, a raktározott késztermék optimális mennyisége stb. Szükség esetén az alkalmazások automatikusan fogadhatnak üzeneteket az adatbázisból,

nemcsak figyelmeztetést az esetleges túlrendelésről, hanem megoldási javaslatokat is a felmerülő problémákra.

Az adatbázisokkal és azok kezelőrendszereivel szemben az egyik legfontosabb követelmény, hogy a lehető legkevesebb felvitt adatból lehessen a lehető legtöbb szükséges információt megszerezni. Ezt a célt annál jobban meg tudjuk közelíteni, minél alaposabban átgondoljuk az adatbázis szerkezetét. A felhasználók hajlamosak arra, hogy az adatbázisban felesleges redundanciát teremtsenek, például amikor valaki az általa használt adatok körébe új jellegű adatokat akar bevinni, mert feladatköre kibővült, és a továbbiakban olyan információkkal is dolgozik majd, amilyenekre előzőleg nem volt szüksége. Ilyenkor mindig célszerű alaposan átnézni az adatbázist, mert ha egy adat vagy adatféleség már megvan a rendszerben, akkor jóval egyszerűbb kiterjeszteni annak hozzáférési jogosultságát, semmint eltűrni a szükségtelen redundanciát.

Az adatokhoz való hozzáférést az adatbázis nyitottságával és a jogosultsági szintek pontos meghatározásával lehet differenciálni. Ez nemcsak azt jelenti, hogy milyen adatokat használhat valaki, hanem azt is, hogy az adatok mely köréhez nem férhet hozzá. A jogosultság vonatkozhat általában bizonyos műveletekre, funkciókra (például, hogy ki számlázhat vagy vehet fel rendelést), de árnyaltabban is meghatározható, hogy ki illetékes valamilyen művelet elvégzésére: csak lekérdezhet, vagy be is vihet adatot; korlátlanul módosíthat egy táblázatot, vagy abban adatok törlésére nincs felhatalmazva stb.).

A vállalatirányítási szoftverek telepítését ezért hosszas tárgyalások előzik meg, hiszen a vállalaton belül a szervezeti egységek feladatait, az illetékességi szinteket, az egyéni és kollektív hozzáférési jogosultságokat a szoftverfejlesztő nem tudja meghatározni.

A vállalatirányitási rendszerben lévő alapalkalmazáshoz kiegészítő részek csatlakoznak, melyek az egyes funkciókat a vállalat egyéni sajátosságaihoz illesztik. Attól függően, hogy milyen mértékben vannak egyéni igények, előfordul, hogy egyszerűbb az alapalkalmazást is módosítani, mint újabb kiegészítő rendszereket csatlakoztatni hozzá. A módosítás főbb lépései:

— Az adatbázist alkalmazó vállalat illetékes szakemberei és vezetői meghatározzák a módosítandó funkciók körét.

— A szoftverfejlesztők ezen információk alapján kidolgozzák, hogy a rendszer szerkezetében milyen változtatások szükségesek.

— A fejlesztők egy nem éles rendszerben, de valós adatokkal módosítják a táblázatokat, új táblákat készítenek, illetve szükség esetén a létrejött új adatkapcsolatoknak, összefüggéseknek megfelelő új alkalmazásokat ültetnek be a rendszerbe. (Sorrendben először az adatbáziskezelőt kell módosítani, utána következik az alkalmazás módosítása, végül pedig sor kerülhet az adatokkal való feltöltésre.)

— A módosított rendszer működését tesztelik, és ha minden rendben van, kezdődhet az új verzió telepítése.

Ilyen módosítási fokozatokon ment keresztül például az Ablesoft rendszer rendelésfelvétellel foglalkozó része, amikor egy megrendelő kérésére a korábban két lépésből — rendelésfelvételből és rendelésjóváhagyásból — álló folyamatot kellett átprogramozni, hogy az tartalmazza többek között a gyárthatósággal, a teljesíthetőséggel, a vevői igazolással és visszaigazolással kapcsolatos adatokat is. Ehhez már a szoftver alaprendszerét és a kapcsolódó alapalkalmazásokat is módosítani kellett.

A módosítások anyagi vonzatát tekintve a fejlesztő cégek a költségeket általában az alaprendszerhez kapcsolódó árakkal együtt és a változtatás bonyolultságától függően kalkulálják. A szerződésben külön pontokban tüntetik fel az egyes változtatásokhoz tartozó munkaórák és egyéb költségek elszámolását, mellérendelve a teljesítés időpontját.

Vállalatok esetében gyakori, hogy földrajzilag külön telephelyeken működnek, viszont szükséges, hogy állandóan „képben legyenek" egymás adatairól. Ez megvalósítható például közös online adatbázissal. Ezt akkor érdemes létrehozni, ha az alkalmazások „erős szerver — gyenge kliens" modell szerint működnek, és a telephelyek a központban lévő szerveren keresztül használják a rendszert. A felhasználó ilyenkor viszonylag egyszerű alkalmazást használ, az utasításokat továbbküldi a központi szerverhez, amely megoldja a feladatot, és az eredményt visszaküldi a telepehelyen dolgozó felhasználóhoz. Az online adatbázis előnye, hogy egy helyen tárolható minden szükséges adat, viszont meg kell teremteni a folyamatos kapcsolatot a telephelyek között.

Amikor a központi szerveren kívül a telephelyeken is van adatbázis, osztott adatbázisokként lehet azokat használni. Ez esetben nem szükséges az állandó online kapcsolat, de az adatcserét meghatározott rendszerességgel ilyenkor is biztosítani kell.

A jól működő rendszereken is változtatásokat követel időnként az adatbáziskezelő programok technikai fejlődése. A fejlesztők általában sebességben, kényelemben, megbízhatóságban ígérnek jobbat a korábbi verzióknál. A legelterjedtebb adatbáziskezelőket (MS SQL, Oracle stb.) általában 3-4 évente újítják meg.

Mielőtt azonban egy vállalat belevágna egy új rendszer vagy egy új verzió telepítésébe, érdemes a fejlesztési szakemberekkel előzetesen konzultálni a váltás előnyeiről, és az érte fizetendő összeg arányosságáról.

Budai Katalin
budai.katalin@simpletech.hu



Integrálás vezeték nélkül

Sybase iAnywhere Wireless Server

A Sybase iAnywhere Solutions kínálatának alapja egy skálázható m-Business platform, kulcsfontosságú integrált eszköze pedig a Wireless Server, amely lehetővé teszi a nagyvállalati adatok és az internetes tartalom mobil és vezeték nélküli alkalmazásokon keresztül történő elérését és sokféle szolgáltatás nyújtását.

A Sybase iAnywhere Wireless Server nyílt szabványokra épülő integrált platformjának vezeték nélküli gateway (átjáró) képessége kibővíti az IP-kapcsolódást olyan csomag alapú hálózatokhoz is, mint az ARDIS, a CDPD és a Mobitex, illetve az áramkör alapú hálózatokhoz, mint a GSM, a GPRS és a CDMA. Az átjáró vezeték nélküli intelligenciát kínál, ezáltal az alkalmazások gyorsabban és megbízhatóbban működnek. Az optimalizált átviteli protokoll (OWTP) csökkenti a kommunikációs többletráfordítást, és tömöríti az adatokat. Az átjáró automatikusan beállítja az átviteli sebességet, alkalmazkodva a változó lefedettségi feltételekhez.

Tartalomszállítás

Az átvitt tartalmat testre kell szabni a mobil eszköz képességeinek megfelelően. A Wireless Server az alkalmazások fejlesztéséhez Java alapú keretben a böngésző által támogatott tartalomtípust szolgáltat, lehetőséget adva egyéni preferenciák definiálására. Több jelölőnyelvet is támogat (HTML, HDML, CHTML, WML, VoiceXML), továbbá az XSL stíluslapnyelv (Extensible Stylesheet Language) segítségével az XML dokumentumokat (Extensible Markup Language) automatikusan átalakítja a specifikus kliens eszközök számára. A Dynamic XML támogatás lehetővé teszi az információ frissítését anélkül, hogy többféle változatot kellene készíteni.

Adatszinkronizáció

Alapkövetelmény az információcsere és a frissítés a távoli eszközök és a nagyvállalati rendszerek között (vállalati adatbázisok, ERP rendszerek, hagyományos rendszerek, üzleti alkalmazások). A helyi adattár gyors hozzáférést tesz lehetővé, az információk pedig olyankor szinkronizálhatók az elsődleges nagyvállalati szerverrel, amikor ez a művelet a felhasználónak leginkább megfelel. A szinkronizálás egyidejű kapcsolódások ezreit jelenti, integrálva a már kipróbált konfliktusfelderítő és konflikusfeloldó technikákkal. Az adatszinkronizációval a mobil dolgozók is idejében hozzájuthatnak az ügyfeleikre vonatkozó információhoz.

Üzenettovábbítás

A mobil üzenettovábbító szolgáltatás lehetővé teszi a kimenő kapcsolati igények sorbaállítását, és a hálózatról érkező üzenetek aszinkron fogadását. A tároló és továbbító üzenetsorok alkalmazása megakadályozza, hogy az üzenetek elvesszenek. Van továbbá nyomógombos üzenettovábbítás, engedélyeztetés és titkosítás. Az üzenettovábbító architektúra a Java Messaging Service (JMS) hálózat- és eszközfüggetlenségét kínálja, és azt a képességet, hogy a felhasználók üzeneteket kaphassanak akkor is, ha a hálózatok között barangolnak (szabadon mozogva és csatlakoztatva egy kábel nélküli adóhoz).

Nagyvállalati integráció

A Wireless Server lehetővé tesz a nyílt nagyvállalati integrációt a meglévő vállalati adatbázisrendszerekkel (Oracle, MS SQL Server, IBM DB2, Sybase Adaptive Server Enterprise stb.). Integrációt kínál a vállalatirányítási (ERP) rendszerekkel (például SAP), üzleti alkalmazásokkal és az üzenettovábbító rendszerekkel. Támogatja a Sybase Enterprise Portallal való teljes integrációt, ugyanis mindkét technológia közös központi architektúrán osztozik, lehetővé téve, hogy a mobil felhasználók személyre szabott információhoz jussanak asztali gépek, kézi eszközök és intelligens telefonok széles skáláját használva. A Java 2 Platform, Enterprise Edition (J2EE) támogatás lehetővé teszi a vállalatok számára, hogy könnyen bővítsék rendszereiket Javában és Enterprise Java Beans-ben (EJB) megírt üzleti logikával. A Sybase iAnywhere Wireless Server egyéb ipari szabványos komponensmodelleket is támogat, mint a CORBA, a COM, a C és a C++, ezáltal egyedi integrációs pontot kínál a heterogén backoffice rendszerekhez. Mindez lehetővé teszi a vállalatok számára, hogy megőrizzék meglévő befektetéseiket, kibővítve jelenlegi infrastruktúrájukat a mobil és vezeték nélküli eszközökre is.

Skálázható alap

A Sybase iAnywhere Wireless Server szimmetrikus multiprocesszoros (SMP) és klaszteres támogatást nyújt több gép, illetve több CPU működtetése esetén. Olyan skálázható környezet hozható létre, amely naponta tranzakciók millióit kezelheti, és egyidejűleg a felhasználók ezreinek rákapcsolódását fogadhatja. A beépített terheléskiegyenlítés és a szerverátkapcsolás révén a forgalom dinamikusan elosztható több rendszer között. A többszálas támogatás lehetővé teszi a fejlett teljesítményhangoló képességek kihasználását (adatbáziskapcsolatok, dinamikus cache, válaszidő csökkentése). A beépített eseménykezelők egyszerűsítik az eseményszintű információhoz való hozzáférést, beleértve a felhasználó által definiált objektumokat is.

Teljes körű biztonság

A rendszer a meglévő biztonsági infrastruktúrával integrálható, miközben a szerver kiemelkedően eleget tesz a vezeték nélküli környezet biztonsági követelményeinek. A titkosítás mellett szerveroldali engedélyeztetéssel ellenőrzi, hogy mely felhasználók jogosultak az információ megtekintésére. Integrálódik a meglévő tűzfallal, a virtuális privát hálózatokkal (VPN) és a PKI technológiával, hogy rugalmas lehetőségeket nyújtson a felhasználók azonosításához és a szerep alapú biztonsághoz. WAP eszközök esetében támogatja a WTLS-t, a kommunikáció titkosítását. Tartalmaz a biztonságkezelő eszközöket, ezzel egyszerűsít olyan feladatokat, mint az SSL támogatás hozzáadása az alkalmazásokhoz.

Koller György
koller.gyorgy@sybase.hu

Üzleti logika a weblapon

A Navision integrált megoldása

Ha egy vállalat bővíti tevékenységét az elektronikus kereskedelem irányába, az új részlegnek összhangban kell működnie a hagyományos kereskedelmi csatornákkal. Az e-boltot valahogy integrálni kell a vállalatirányítási rendszerbe. A kis- és középvállalkozások vállalatirányítási rendszerét és e-kereskedelmi megoldásait fejlesztő Navisoft cég Navision Software terméke ezt példázza.

Az elektronikus keresés és vásárlás a vevő szemszögéből nézve úgy zajlik le, hogy online kapcsolaton keresztül megtekinti az eladó cég árukínálatát és árait. Az árukat jelképesen a bevásárlókosarába helyezve ajánlatot kérhet az eladótól, vagy rögtön megrendelheti a kiválasztott holmit. Korábbi rendeléseit később bármikor megtekintheti, az aktuálisakat átnézheti és módosíthatja.

Az eladó szemszögéből az értékesítés folyamata hasonló, mintha „az utcáról betérő" vevő kiszolgálásával kellene foglalkoznia: ajánlatkészítés, rendelésfeldolgozás, számlázás, a vevő folyószámlájának kezelése — mindez a hagyományos rendszer üzleti logikáját követve.

Testre szabás

A Navision Web Shop az e-boltok sajátosságaira specializálódott, és a rendszergazda a később kialakuló üzleti megoldásokat is könnyedén beillesztheti abba. Az üzleti élet nemzetközivé válására tekintettel az egyes cikkekre vonatkozóan több nyelven is rögzített adatokat a Navision Financials kezelni tudja, ezért nem okoz gondot az idegen nyelvű portálok generálása. A Web Shop integrálása a Navision Financials rendszerrel természetesen az alapalkalmazást is érinti, hiszen a Web Shop telepítése változást okoz az alkalmazás felhasználói felületén. Ezek jellemzően a vevők nyilvántartásánál következnek be.

A kisebb-nagyobb változások gördülékenyebbé teszik az értékesítési és kapcsolattartási folyamatokat. Például minden vevőre külön-külön meghatározhatjuk, hogy az e-boltban feladott rendelés esetén automatikusan megtörténjen-e a készletből való tényleges foglalás, vagy csak a visszaigazolás után. További lehetőség olyan prioritás érvényesítése, hogy a rendszeresen nagy értékű megrendelést adó ügyfelek kiszolgálása automatikusan elsőbbséget élvezzen a kisebb megrendelőkével szemben.

A Web Shop telepítéséhez helyi fejlesztési lehetőségek is kapcsolódnak, például új titkosítási funkciókat tartalmazó DLL alkalmazható, amelynek függvényei egyaránt hívhatók VB-Script vagy a Navision C/AL kódból. A rendszergazda meg tudja változtatni a honlap struktúráját, hogy saját ötleteivel is segítse a vevők eligazodását. A rendszer támogatja tetszőleges lapok megjelenítését, így hivatkozni lehet a műszaki leírásokra, ha pedig azok elérési helye megváltozik, könnyedén követhető a változás. A lapok integritásának ellenőrzését segítő funkció figyelmezteti a rendszergazdát a nem jól megadott hivatkozásokra, így azok kijavíthatók, mielőtt még a lapokat feltöltenék a webszerverre. A testreszabás egyik módja az üzleti logika megváltoztatása Navision C/AL kódok írásával. A másik az ASP kódok átírása, amikor csak a weblapokat szeretnék módosítani. A különböző megjelenésű lapok így azonos üzleti logikát használhatnak.

Teljesítménymérce

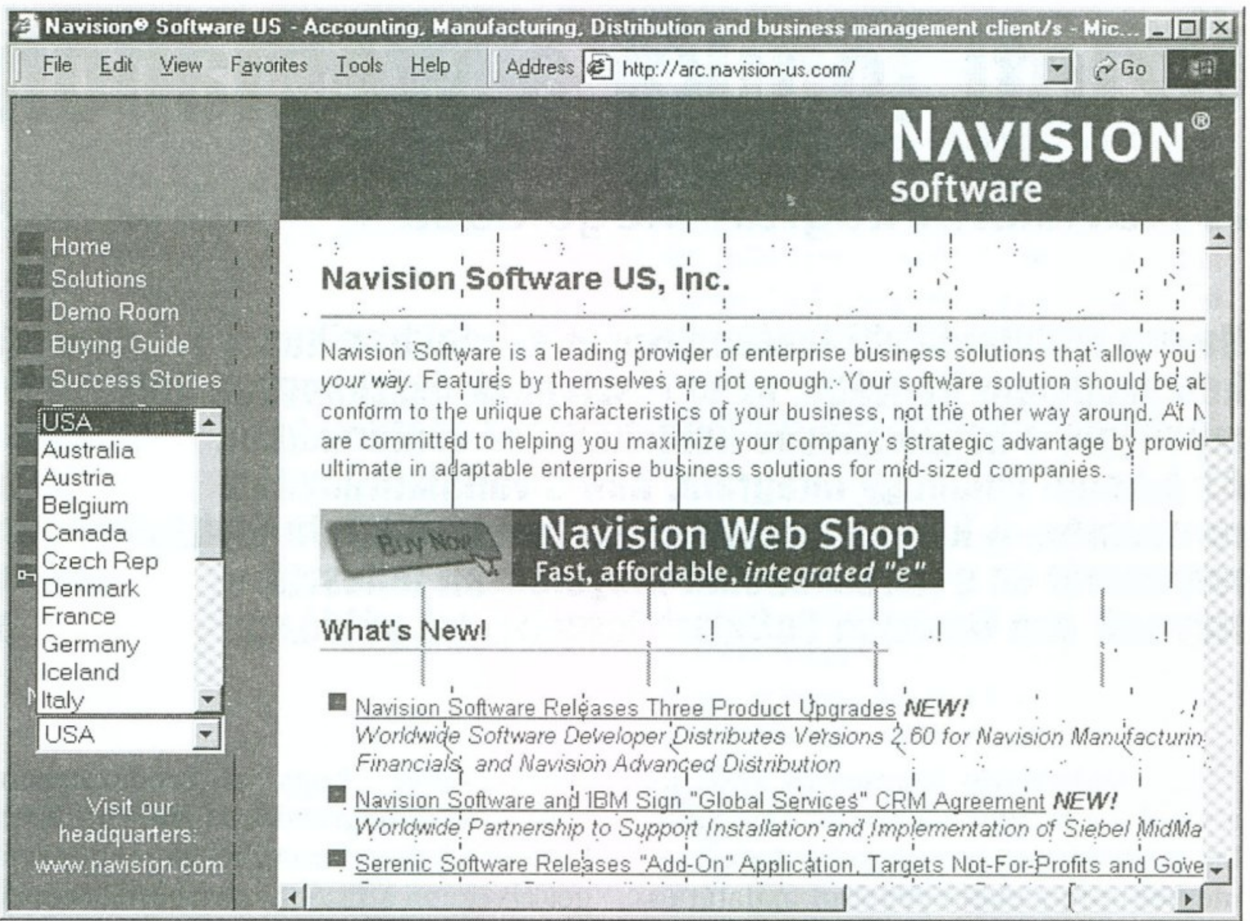
A rendszerek teljesítményét jelentősen befolyásolja, hogy mennyire használják ki az architektúra lehetőségeit. Jelen esetben ez azt jelenti, hogy minden egyes rétegének megvan a külön feladata a vásárlási kapcsolatokban. Például a böngészőn megjelenő képek egyszeri átvitele növeli a teljesítményt. A letöltött képek és tárolási helyük ismerete alapján elérhető, hogy minimálisra csökkenjen az adatmozgatás a böngésző és az adatbázis között.

A Navision Financials adatbázisa (Navision szerver) terhelésének csökkentésére beiktatott köztes rétegben a Microsoft SQL Server számtalan átmeneti táblát tárol, és szükség szerint szinkronizál a Navision szerveren lévő adatbázissal. Azért esett a választás az SQL Serverre, mert a tárolt táblák száma egy átlagos látogatottságú e-boltban is elég nagy lehet. A táblák kapcsolatai is elég összetettek, emiatt széles skálázhatóságra van szükség. Mivel az

Termék	Felhasználás
Messagge Queue Server	A böngésző és a Navision szerver közti adatcsere lebonyolítására az üzenetküldő API segítségével
Site Server, Commerce Edition	A böngészőn megjelenő lapok és az azokon megjelenő linkek generálására. A lapokon a munkafolyamatok végrehajtása és a munkafolyamatok statisztikájának felépítése a feladata
SQL Server	A Navision szerver és a böngésző közti adatcseréhez szükséges átmeneti táblák és adatok tárolására
Transaction Server	A teljes tranzakciós integritást igénylő műveletek lebonyolítására

SQL Servert az architektúra részeként gyakorlatilag átmeneti tárolóként szolgál a böngésző és a Navision szerver adatbázisa között, a Navision adatbázisra közvetlenül kapcsolódó dolgozók semmiféle sebességcsökkenést nem érzékelnek a Navision Web Shop által kezelt e-bolt megnyitása után.

A teljesítmény növelését szolgálja az a megoldás is, hogy minél ritkábban kelljen felvenni a kapcsolatot a Web Shop és a Navision Financials rendszer között. Gondoljunk csak bele, hogy az e-bolt vásárlóinak különféle igényei — például e-bevásárlókosaruk tartalmának újraszámítása vagy a raktárkészletben való keresés — milyen mértékben terhelné a Navision szervert. A Navision Web Shop architektúrájában a Microsoft SQL Server beiktatása révén ehhez általában nincs szükség a Navision szervereren lévő Navision adatbázis eléréséhez (vagy csak olvasási joggal, ami nem okoz rekordzárolást). Ezáltal nemcsak a szerver teljesítménye nő, hanem csökken a hálózati forgalom is.



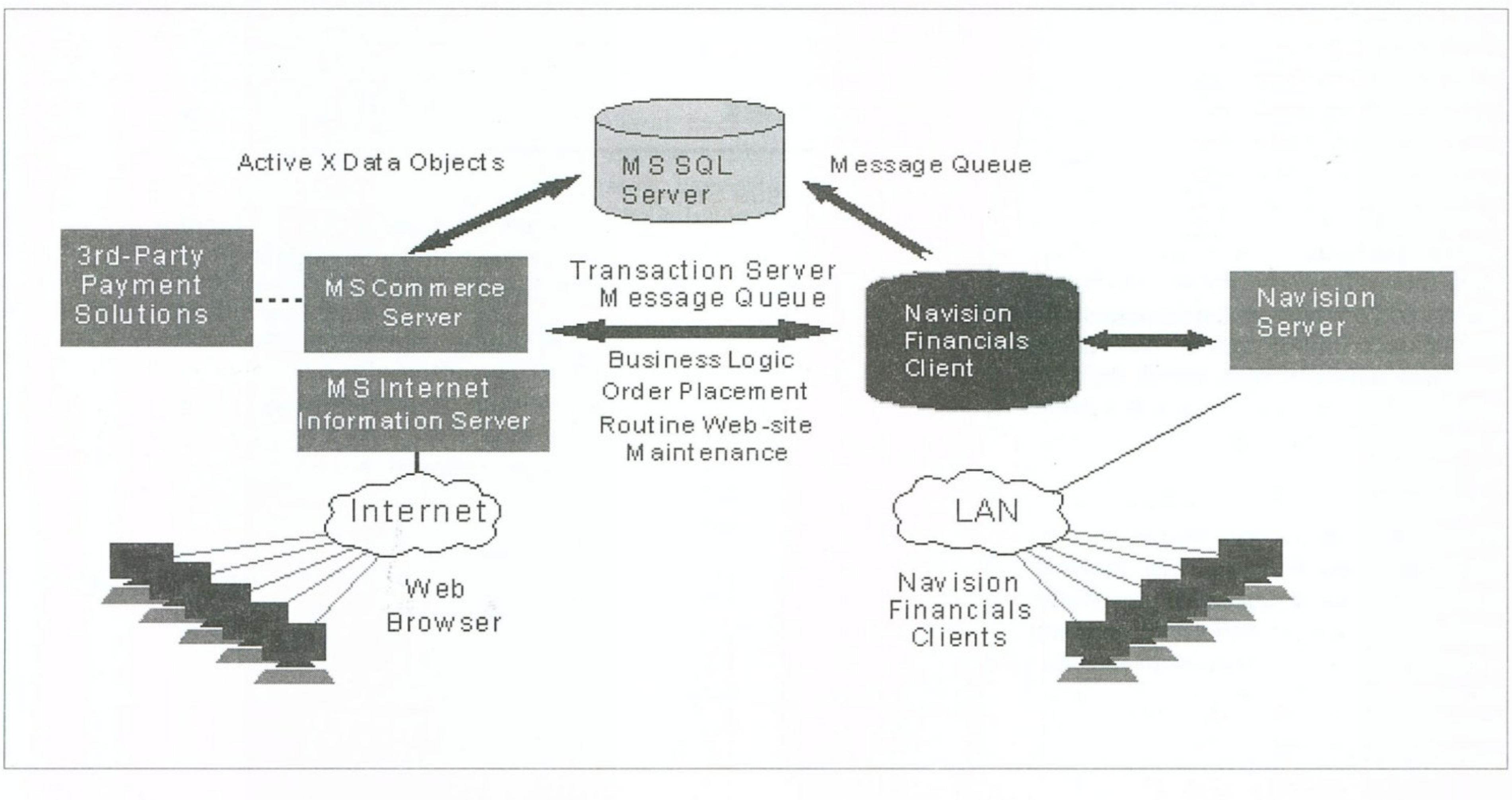
Biztonsági felszerelések

Természetesen minden számviteli és kereskedelmi rendszer egyik kulcskérdése a biztonság. Az e-boltba betérő és a Navision szerverről adatot igénylő felhasználónak rendelkeznie kell érvényes azonosítóval és kódszóval. Ezeket maga a Navision szerver is generálhatja, amikor a vásárló feliratkozik. A generált felhasználói azonosítót és kódot a vevő e-mailben kapja meg. A Navision szerver által generált felhasználói azonosítók a Navision Financials jogosultsági rendszerében a rendszergazda által kijelölt jogosultságokkal bírnak. A Web Shop felhasználóinak mozgása is nyomon követhető, csakúgy, mint a Navision adatbázist közvetlenül elérő felhasználóké.

Az alkalmazott N rétegű architektúra is a Navision adatbázis biztonságát szolgálja. Minden Web Shophoz kapcsolódó tevékenység, amely a Navision szerverrel történő kommunikációt igényli, előírt formátumú üzenetküldő API-n keresztül zajlik. Azt az API határozza meg, hogy egy Web Shop felhasználója mit tehet meg (még ha esetleg teljes jogosultsággal is rendelkezik a Navision adatbázis felett). A kommunikáció biztonságáról a Microsoft Site Server által támogatott SSL (Secure Socket Layer) gondoskodik. A böngésző és a Navision Web Shop között áramló minden adat titkosításra kerül.

A Navision Web Shop és a kiszolgáló architektúra Microsoft termékekre épül. Az e-bolt üzemeltetője robusztus és skálázható infrastruktúrához jut. Mindezek mellett maga a Navision szerver nemcsak Microsoft platformon futhat, hanem AIX, HP-UX vagy Reliant Unix rendszereken is.

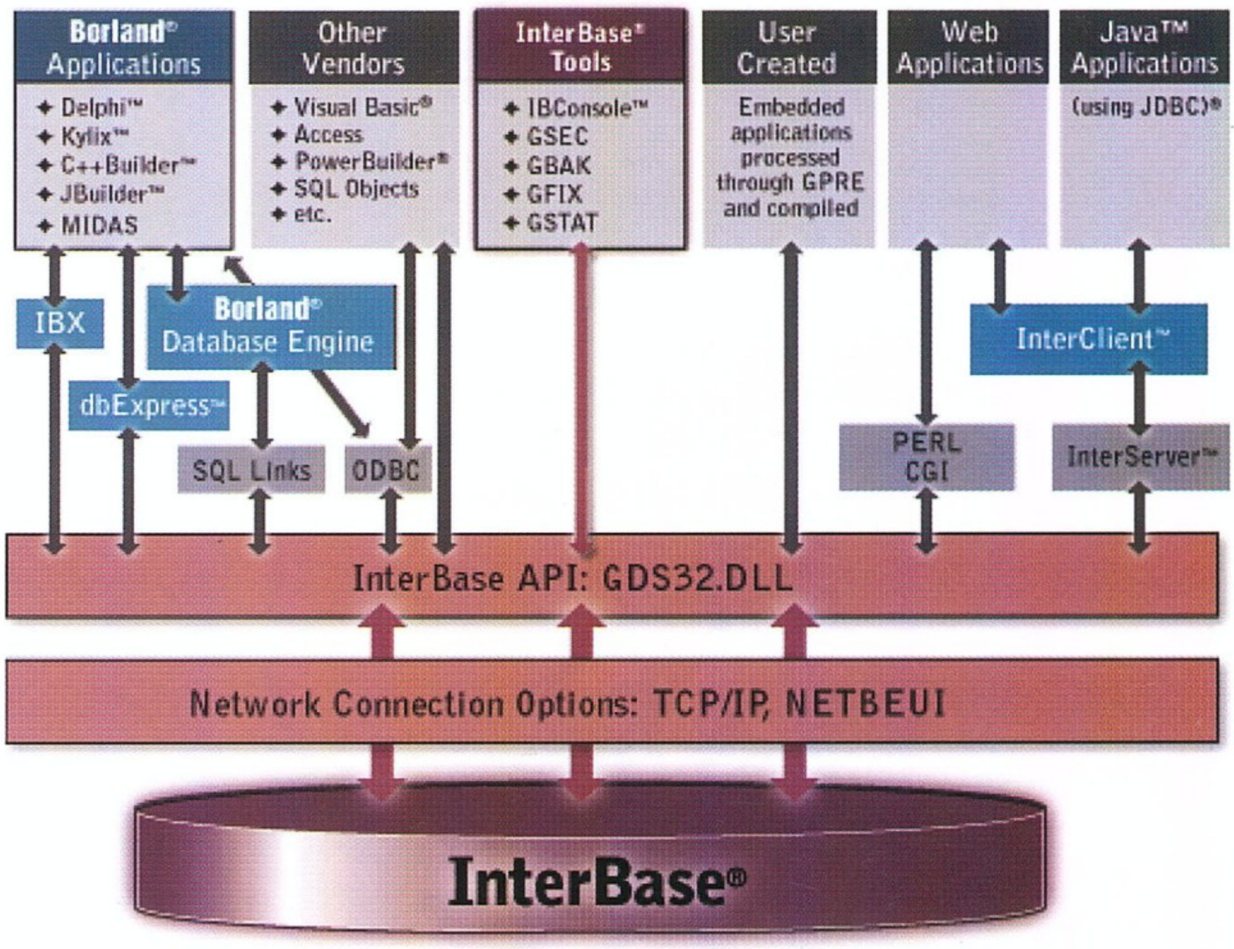
Borsics János
borsicsj@multisoft.hu

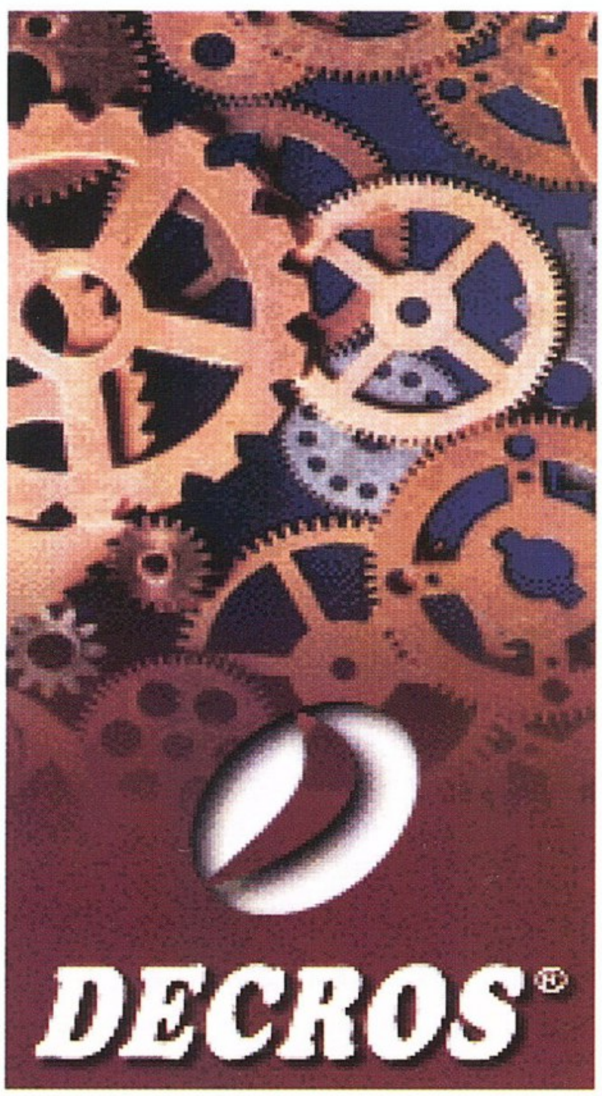


Adatbázis

Az „Adatáradat" témájához kapcsolódva a CD-mellékleten több adatbázis-kezelő rendszert is közreadunk. Amint arról korábban már beszámoltunk, a Borland által gondozott InterBase-t a 6-os verziótól kezdődően a Borland nyílt forráskódúvá tette (http://www.borland.com/devsupport/interbase/opensource). Ennek megfelelően a futtatható programhoz is szabadon hozzájuthatnak azok, akik nem akarják a kódot saját maguk lefordítani. A bináris állományok csomagja tartalmazza a 32 bites Windowshoz készült szerveroldali és kliensoldali InterBase 6.01 alkalmazást. A ZIP-archívumban a telepítőkészlet mellett programozási eszközkészletet is találnak (SDK, Software Development Kit). A futtatható állományokat tartalmazó Linux-csomag kétféle szerverkiszerelése közül az egyik csatlakoztatott kliensenként egy processz futtatását teszi lehetővé, a másik a szuperszerver változat, amely egy szerverprocessz elérését több felhasználó számára több szálon képes biztosítani. Mindkét csomag a disztribúcióktól kevésbé függő tar archív formában található, a telepítőkészletet a .tgz állomány tartalmazza, a tényleges telepítést kibontás után az install szkriptfájl végzi el.

Az InterBase egyik előnye, hogy a Borland jávás fejlesztéseinek köszönhetően Linuxra és Windowsra nyílt forráskódú kliensprgram is készült hozzá, az InterClient. Ezt is megtalálják CD-mellékletünkön. Mindazonáltal figyelembe kell venni, hogy inkább kisérleti fejlesztésű és nem éppen friss produktumról van szó. Tekintettel azonban arra, hogy a forrásállomány publikus, bizonyára sokak érdeklődését felkelti, Java programozási tanfolyamunkat pedig jól kiegészíti.





Más forráskódokat is hozzáférhetővé tettünk a CD-n az adatbáziskészítéshez, illetve azoknak, akik egyszerűen csak kíváncsiak egy ilyen rendszer belső felépítésére. A CVS rendszerben tárolt naprakész források egyébként a http://sourceforge.net/projects/interbase cím alatt érhetőek el, a licenc továbbra is a Mozilla modellt követi (Mozilla Public License, MPL), a rendszerplatformok közül a Windowst (95/98/2000) és Posixot támogatja, programozási nyelve a C. Az ftp.borland.com webhelyről letölthető állományok CD-mellékletünkön is megtalálhatók, az említett forrásadatbázisból pedig anonim módon a „cvs -d:pserver:anonymous@cvs.Interbase.sourceforge.net:/cvsroot/interbase login" és a „cvs -z3 -d:pserver:anonymous@cvs.Interbase.sourceforge.net:/cvsroot/interbase co modulnevek" parancsokkal juthatunk hozzá.

A szabad forráskódban fejlesztőknek közreadunk a közelmúltban felfrissített két adatbáziskezelőt. A MySQL és a PostgreSQL az SQL-nek forráskódban is elérhető implementációja, szabad az útjuk a különböző platformok felé.

Felkeltheti a programozók érdeklődését a Combit cég (www.combit.de) List&Label programjának próbaváltozata, illetve az annak képességeit bemutató program. Az adatbáziskezelés végtermékének is tekinthető jelentések készítésekor használható segédeszközről lapunk 53. oldalán olvashatnak, az említett anyagokat tartalmazó önkicsomagoló fájlok pedig a CD Lapraforgó rovatában találhatók meg.

Védekezzünk

A CD-mellékleten rendszeresen közölt antivírus programok szükségességéről „Vírusőrjárat" rovatunk alapján bárki tájékozódhat. A vírusok elleni védekezés jegyében tesszük közzé a korábban AVP néven ismert program utódjának, a Kaspersky AntiVirus (KAV) legfrissebb verziójának publikus változatát, amelyet 30 napig kipróbálhatunk, ezalatt érvényes az online frissítési lehetőség is.

Nem a fájlokat, hanem a felhasználók idegeit károsítják a webes böngészés során eléjük tolakodó reklámablakok. Az ilyen zaklatás elleni védekezést hivatott segíteni a Webwasher program. A CD-n közreadott egygépes változat szabadon használható 32 bites Windowson, Linuxon és Macintoshon. A három platformra összesen alig több mint 2 MB-os csomagot a Vendégoldal rovatában találják meg.

Ugyanebben a rovatban adunk közre a Decros biztonsági programcsomagot, valamint az immár hagyományosan egy kisebb válogatást a Microsoft javítókészleteiből.

Simay Endre István


telnet
TELNET PRODUKCIÓ
WWW.STOP.HU
UGORJ BE...
ÉS NYERJ!
NAPONTA WAP-TELEFONT
FŐNYEREMÉNY: DIGITÁLIS KAMERA
stop!
www.stop.hu

Egérszabadság

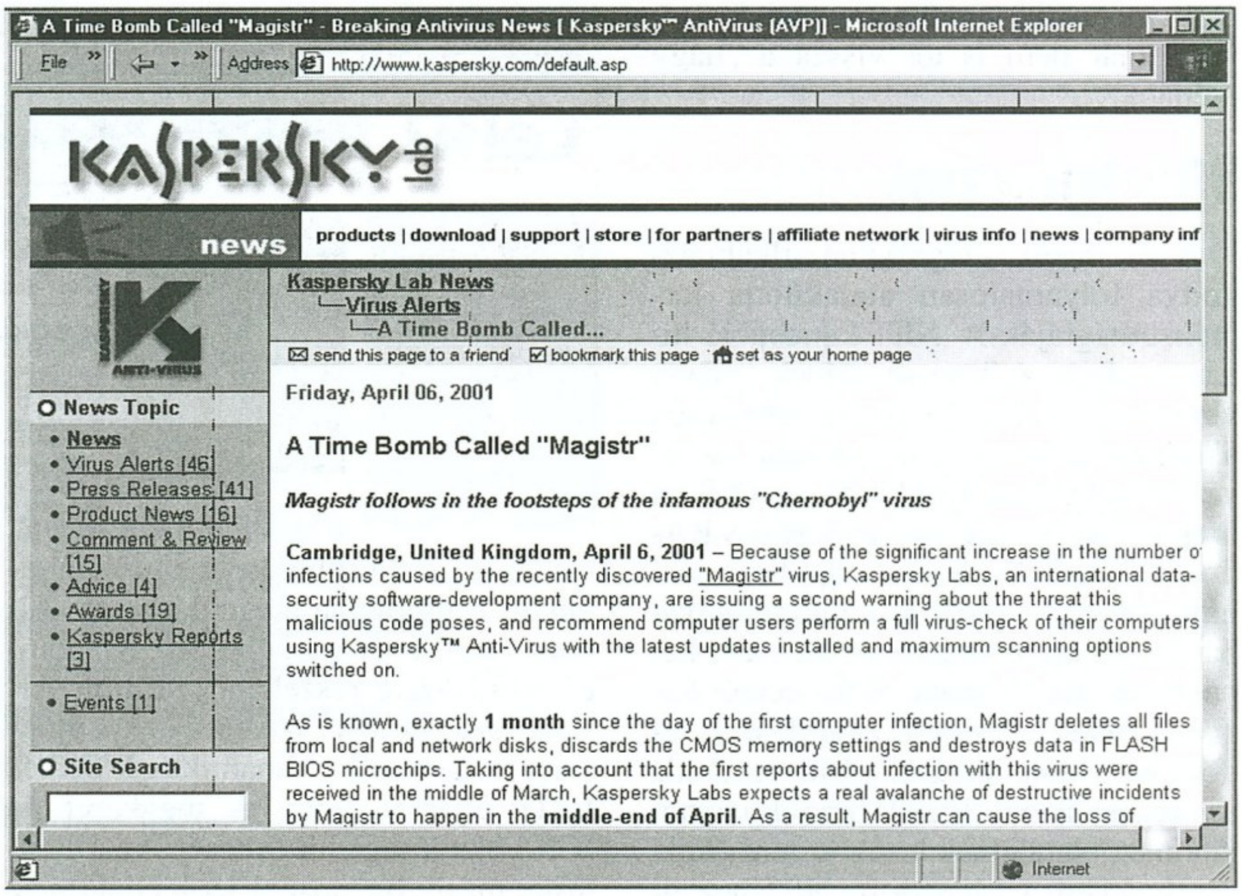
A billentyűzet és az egér zsinórja minden PC-használónak okoz egy kis kényelmetlenséget és kötöttséget, de a legtöbben úgy gondolják, hogy ennyit igazán el lehet viselni a két legolcsóbb periféria használatáért. Az igényesebbek persze választhatnak. A Logitech elég korán elkezdte a drótnélküli eszközök (köztük a játékvezérlők) fejlesztését — azoknak, akiket zavar a munkaasztalon kígyózó, időnként átrendezésre kényszerítő, valamibe mindig beleakadó és nem mindig elég hosszú póráz. A CeBIT-en bemutatott legújabb optikai egér (Cordless MouseMan Optical) az Agilent optikai pozicionáló technológiájával érzékeli az egértest relatív mozgását, és a helyzetváltozásokat mini rádióadó közvetíti a PC-hez kapcsolt vevőhöz.

SuSE-szerverek

A kereskedelmi csatornákon értékesített Linux disztribúciók elterjedtsége nem teljesen azonos az ingyenesen, de terméktámogatás nélkül elérhető változatokéval. A http://linux.box.sk felmérése szerint a SuSE például az ingyenes kategóriában „csak" dobogós helyen szerepel, ugyanakkor a CeBIT-en közreadott statisztikák szerint a terméktámogatásos, dobozos verziók értékesítését tekintve már nemcsak Európában, hanem Amerikában is piacvezető lett. A CeBIT-en bemutatták a SuSE 7.1 PowerPC-s változatát, továbbá megjelent a vállalati ügyfélkörnek több szervercsomag is: a Linux Business Server, a Linux Enterprise platform és a Linux eMail Server II. Ezekbe még a jól bevált 2.2.16-os kernelt építették be.

Új vandál a porondon

A többnyire csak bosszúságot, például hálózati forgalomtorlódást okozó mostani féregvírusok között időnként akadnak olyanok is, amelyek romboló rutinokat tartalmaznak. Április elején bukkant fel egy ilyen gonosztevő, a szintén levélmellékletben terjedő Magistr féreg (W32.Magistr.24876@mm, Worm.Magistr, PE_MAGISTR.A). A fertőzést követően egy hónapig szunnyad a rendszerben, utána elkezdi a fájlok törlését. Miként arra a Symantec (http://www.symantec.com/avcenter) hírlevele felhívta a figyelmet, ezért is célszerű bizonyos időnként lefuttatni a frissített antivírus programot. A Kaspersky Labs (http://www.kaspersky.com) elemzése szerint a Magistr törli a fájlokat a helyi és a hálózati meghajtókról, sőt törli a CMOS-t és a Flash BIOS chipek tartalmát is, feledhetetlen izgalmakat okozva az amnéziába eső gépek gazdáinak.



Komplex szűrés

Az interneten való böngészés során szinte mindenkit zavaró (és 1% alatti rákattintási arányú!) reklámcsíkok és reklámablakok kiszűrése különösen azoknak fontos, akik sávszélességét ez a dekoráció túlságosan megterhelné, vagy akikben nehezen alakul ki az immunitás, a megfelelő szelektív vakság. A szűrőeszközök fejlesztői közül a Webwasher.Com március elején jelentette meg a böngészést szűréssel gyorsítani hivatott program Windows után Linux platformra is megírt 3-as verzióját. Az egyéni felhasználók ingyenesen letölthetik azt a cég weblapjáról, de megtalálják mostani CD-mellékletünkön is. A fejlesztők a telepítés ellenőrzésére külön oldalt rendeztek be (http://www.webwasher.com/en/products/wwash/testpag1.htm). A vállalatok számára elkészült a rendszerhez illeszthető, szerver alapú Enterprise Edition változat. A helyi hálózatok adatforgalmának ellenőrzésére ebbe már integrálták a DynaBLocator dinamikus URL adatbázist is, amely felhasználja a Cobion képfelismerő technológiát.

Cégen belüli ASP

Az alkalmazásszolgáltatás (ASP, Application Service Provider) egyes előrejelzések szerint egyre népszerűbb lesz, mert a cégek számára feleslegessé teszi a ritkán használt szoftverek beszerzését. Távoli ASP igénybevétele esetén viszont bizalmi és biztonságtechnikai aggályok merülhetnek fel, hiszen sokan a levelezésüket sem szeretnék mások által hozzáférhető külső szerveren tárolni. Innen jött az ötlet, hogy az eredetileg internetes ASP technológia a cégeken belül kialakított intraneten is alkalmazható, például az önálló elszámolási egységekből álló cégek hálózataiban a szoftverek használatának ellenőrzött és mért megosztására. Ilyen rendszerben vizsgázott a Revolution Zenit SQL, amikor a Group 4 Securitas rendszerébe integrálták. A központi rendszer alkalmazásszolgáltatóként elégíti ki a helyi felhasználói igényeket és a testvércégtől becsatlakozókat.

Beépülő BeOS

A sokáig nagy reményeket tápláló BeOS operációs rendszer házatáján a közelmúltban lényeges irányváltást sejtető változások zajlottak le. A fejlesztés fókuszába az internetes eszközök (IA, internet appliance) kialakítását helyezték. A BeIA, mely a BeOS alapján jött létre, tulajdonképpen nem más, mint egy testre szabható böngészőfelület, mely elfedi a mögötte működő rendszert, és teljes funkcionalitását alárendeli az internetes multimédia kiszolgálásának. Ehhez sikerült olyan támogatókat is megnyernie, mint a Sony. A nagy kérdés persze az, hogy az internet konjunktúrájának oldalvizein a BeOS mennyire evez majd sikeresen. És az sem mellékes, hogy a Windows uralma

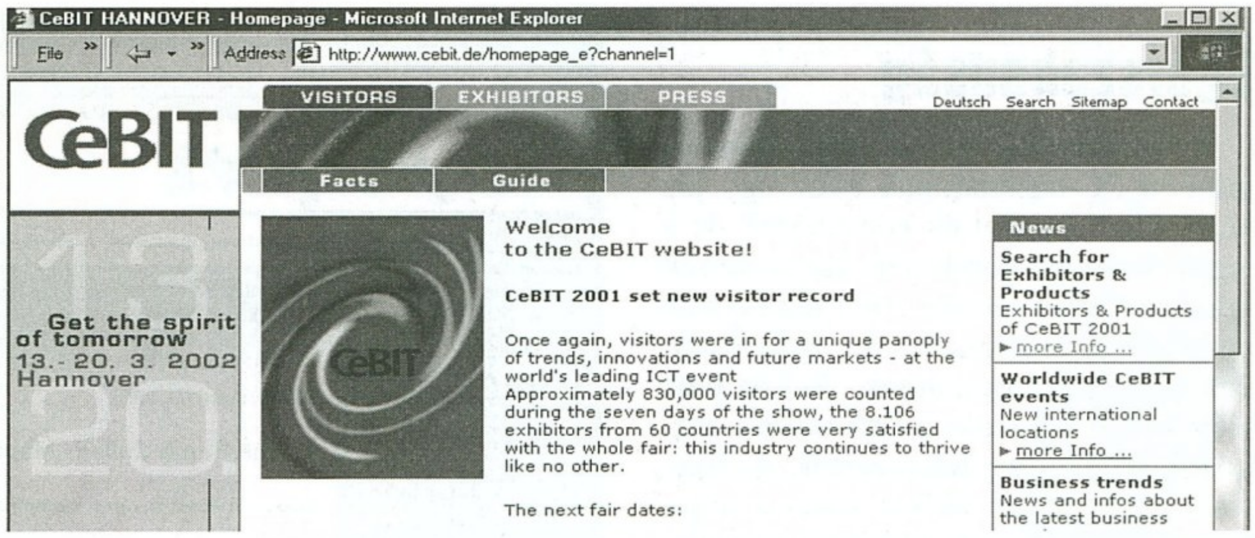
előli menekülés csak átmeneti kitérő, vagy már nem is tér vissza a „nagy arénába”.

VSA-fejlesztés

A Microsoft az internethez alkalmazkodva folyamatosan átalakította termékstratégiáját. A .NET koncepció kihatott fejlesztőkörnyezetének újabb verzióira is, megjelent az internetes alkalmazásfejlesztés támogatása, a Visual Studio for Applications (VSA), melynek segítségével a jelenlegi VBA-hoz (Visual Basic for Applications) hasonló elvek szerint az internet irányában is bővíthetők egy-egy program képességei. Az új rendszer licencelői közül a Scala az elsők között volt, és döntött a VSA integrálásáról az iScala termékcsaládba. Ezzel termékpalettáján megjelenhetnek azok a termékek, amelyekben a Microsoft .NET-hez illeszkedő alkalmazások készülhetnek.

Dinamikus VPN

A virtuális magánhálózatok (VPN, Virtual Private Network) az internet technológiáján, illetve annak kapcsolati protokollján (IP) alapuló munkavégzésben egyre nagyobb szerepet játszanak. A VPN-re csatlakozni kívánó munkaállomás azonosítása tulajdonképpen az IP-címmel történik, a távoli eléréshez használt modemes, ISDN és ADSL alapú kapcsolatok során az IP-cím azonban rendszerint véletlenszerűen kerül kiosztásra, a szolgáltató a DHCP protokoll segítségével dinamikus IP-címmel látja el a felhasználót. Ez azt jelenti, hogy a statikus IP-címre definiált VPN-alkalmazások elveszíthetik biztonságosságukat. A probléma megoldására az Allied Telesyn (www.alliedtelesyn.com) az AlliedWare részeként szállított AR útválasztó (router) számára olyan módszert dolgozott ki (dinamikus IPSec és L2TP eljárás), hogy az megkapta az ICSA biztonsági minősítését. Az AlliedWare másik biztonsági szolgáltatása az operációs rendszerbe integrált Nemesis Stateful Inspection Firewall.



Cáfol a Microsoft

A Der Spiegel online kiadásában megjelent egy hír, majd jött annak cáfolata. A Microsoft által kiadott közlemény idézi a német honvédelmi minisztérium szóvivőjét: „Téves az az állítás, hogy a német hadsereg a jövőben nem kíván Microsoft szoftvereket alkalmazni a biztonsági szempontból kényes területeken”. Érdemes azonban figyelmesen elolvasni a nyilatkozat másik felét is, amelyben közlik, hogy a hadseregnél a titkos információk védelmét egyrészt egy tűzfal szolgálja, másrészt a titkosítás, és ezek az alkalmazott szoftverektől függetlenül működnek. („These protection functions work independently from the software used.”) Hát igen! A német honvédelmi minisztériumnak meg kellett nyugtatnia a Microsoftot... de a németeket is!

„Viszonylag...”

A két nagy hazai számítástechnikai kiállítás, az Info és Compfair a cégek távolmaradásával küzd, a hannoveri CeBIT mint a szakma csúcsrendezvénye viszont idén ellenkező előjelű rekordokat döntött meg. Összesen 422 ezer m^2 nettó területen 60 országból 8106 cég állított ki, a látogatók száma pedig meghaladta a 830 ezret, miközben a rendezők Hannover városa az elviselhető egyhetes „teherbíróképességet” 700 ezerre taksálja. (Részletesebb adatok: http://www.cebit.de.) Az informatikai piac Amerikában bekövetkezett visszaesése az idei CeBIT-et tehát még nem érintette, de számíthatunk rá, hogy a hullám(völgy) egy kis fáziseltolódással Európát is eléri.

A kiállítók rengeteg újdonságot vonultattak fel, de megfigyelhető volt egy furcsa ellentmondás. Miközben az informatikai világ a számítástechnika és a távközlés konvergenciájára, az adatkommunikációra, az ehhez szükséges technológiákra koncentrál, a CeBIT-en felvonultatott eszközök tetemes hányada inkább bazári képzeteket ébresztett. Joggal jellemezte a Wall Street Journal CeBIT-tel foglalkozó cikkének címe a kiállítást úgy, hogy „Gadgets, gadgets, gadgets”. A megfelelő magyar kifejezések közül egyik sem túl hízelgő: bigyó, kütyü, mütyűr és ehhez hasonlók, de jelentése felől a Webster szótár meghatározása sem hagy kétséget, főleg ha az első helyen szereplő „apró mechanikai eszköz” után elolvassuk a másodikat is: „valamilyen érdekes, de viszonylag haszontalan vagy szükségtelen tárgy” (any interesting but relatively useless or unnecessary object). Hát itt tartunk 2001-ben! Hitték volna?

SMS

✉ A Hewlett-Packard bemutatta első kombinált meghajtóját, amely a CD-írás mellett DVD-lemezeket is olvas.

✉ Több mint 80 előadás hangzott el Balatonfüreden a Decus idei konferenciáján, amely korábban a Digital „híveinek”, most pedig a Compaq felhasználóinak fontos szakmai fóruma.

✉ Az Intel megjelentette a SpeedStep technológiát alkalmazó 1 GHz-es mobil Pentium III processzort, és a Compaq új Armada és Presario táskagépei már ezzel lesznek felszerelve.

✉ A PSINet Magyarország az anyacég tőzsdei árfolyamának csökkenése ellenére változatlanul folytatja tevékenységét.

✉ A Multi-Project Wafer (MPW) nevű akcióterv keretében az IBM kis méretű és elérhető árú szilicium–germánium (SiGe) chipeket akar kifejleszteni cégek és egyetemek kutatóinak bevonásával.

✉ A BME Pénzügyi Informatika Kutatóközpontjának (PIKK) informatikai laboratóriumában virtuális bemutatóterem nyílt a Graphisoft GDL geometriai leírónyelv megismertetése érdekében.

✉ Az Avaya a kábelhálózatok felügyeleti lehetőségeinek kibővítésére elkészítette az iPatch rendszert.

✉ A Java technológián alapuló mobiltelefonok terjedését jelzi a Sun, a Nextel és a Motorola együttműködése a Motorola i85s és i50sx készülékeinek forgalomba hozatala kapcsán.

✉ A magyar Noreg kapta meg idén a térség legdinamikusabban fejlődő ISS képviseletének járó díjat, ami elismeri az ISS adaptív biztonsági programcsomaggal Magyarországon elért sikereket.

Saját erőből

A kibontakozás forgatókönyve

„Virtuális dátum" című múlt havi írásomban már felvetettem, hogy egy összevont számmal — vagy két nyári szám kihagyásával — ismét vissza tudnánk térni a hóeleji megjelenési ritmushoz. A körülményeket időközben tovább mérlegelve arra a következtetésre jutottunk, hogy ez a „lépésváltás" egyúttal alkalmas lenne arra is, hogy teljesítsük néhány korábbi ígéretünket. A szükségmegoldásból fokozatosan kikerekedett egy sokoldalú, előremutató terv, amelynek gondolatmenetére remélhetőleg olvasóink is ráhangolódnak, megvalósulásával pedig elégedettek lesznek.

Az új Alaplap nem akármilyen jubileumhoz közeledik. A mostani szám a 197., a májusi lesz a 198., a júniusi a 199. és a júliusi *lenne* a 200. szám. Magyarországon még egyetlen informatikai havi folyóirat sem érte el ezt a kerek számot, illene ehhez méltóan megünnepelni. De éppen júliusban és augusztusban? Kevés kivételtől eltekintve a számítástechnikai cégek ilyenkor „kampánycsendet" tartanak, az olvasók jelentős hányada pedig nyaral, és csak szeptemberben dolgozza fel olvasási restanciáját. A 199. júniusi szám után ezért tartunk egy lélegzetvételnyi szünetet, és utána jöhet a 200., a szeptemberi szám. A júliusi és augusztusi szám kimaradása esetén a szeptemberi megjelenés már visszaállhat a hónap elejére. A további ütemterv ehhez igazodhat, csak azt be is kell tartani.

Hogyan viselik el olvasóink a két nyári szám kimaradását? Bízom benne, hogy nagyon fog nekik hiányozni. Persze a tényleges kihagyás csupán egy hónapnyi, mert június végétől augusztusig kapható lesz a júniusi szám, szeptembertől pedig már a szeptemberi. A hírlapárusoknál vásárlók egyébként is „egyik lapról a másikra élnek", az előfizetőket pedig nem éri hátrány, 2 hónappal meghosszabbodik előfizetésük érvényessége, mert éves előfizetésük — akár teljes naptári évre fizettek elő, akár évközi rendelés volt — 12 számra vonatkozott.

A 200. számhoz igyekszünk sok érdekes anyagot összegyűjteni, és akkorra időzítjük az 1991 óta minden páratlan évben lebonyolított olvasói közvéleménykutatásunkat is, sok értékes nyereményt sorsolva ki a terjedelmes kérdőív kitöltői között.

*

Elég régen megígértük már, hogy összegyűjtjük és CD-n megjelentetjük a lapunkban eddigi közölt írásokat. Nem azért késünk vele, mert nem foglalkozunk a feladattal, hanem mert sokkal nagyobb munkának bizonyult, mint amilyennek eredetileg gondoltuk. Minden maradék időt kihasználva régóta dolgozunk rajta, és most már félig-meddig elő van készítve az 1990–2000 közötti 10 év anyaga. Még ebből a digitalizált periódusból is elég sok cikket kellett szkennerrel beolvastatni, és a digitálisan meglévő korai cikkekből is csak rengeteg manuális munkával és egyedi „újraszerkesztéssel" tudtunk egységes szövegformát kialakítani. Összességében mintegy 3000 cikkről van szó... A 200. szám alkalmából azonban szeretnénk végre tényleg kirukkolni vele, annál is inkább, mert jelen pillanatban ez ígérkezik legbiztosabb bevételi többletforrásunknak.

*

Másik nagy adósságunk olvasóinkkal szemben weblapunk megújítása. Talán nem feltételezik rólunk, hogy nem tudnánk gyorsan „átpofozni" egy kicsit azt a három éve változatlan formájú és szerkezetű, meglehetősen statikus honlapot. Az igazi kérdés egyáltalán nem ez, hanem hogy mivé, milyenné... Mert készíthetnénk ugyan egy másik „faliújságot" a megjelenő cikkek egy részét is felhasználva, de a hozzánk érkezett vélemények arról tanúskodnak, hogy olvasóink jelentős része ennél jóval többet vár tőlünk, a laphoz képest valami többletet, és ha már változtatunk, akkor legyen élő, friss tartalomszolgáltatásunk is. Nem árt azonban emlékezni arra, hogy a kezdeti eufória elmúltával milyen sok hasonló próbálkozás fulladt kudarcba szerte a világon, pedig azoknak volt befektetendő felesleges tőkéjük. A weblap átalakítására azonban mindenképpen keresünk majd valamilyen egészséges kompromisszumot, és ugyancsak a 200. szám megjelenése kapcsán a www.alaplap.hu új arculatot ölt majd, és új tartalommal telik meg.

*

A lap egészére vonatkozóan jelképes „előremenekülésünk" egy vonatkozásban egészen konkrét formát is ölt: a Dózsa György úti (volt szakszervezeti) irodaházat, ahol 3 éve lakunk, az új tulajdonos most átépíti, a bérlőket ennek megfelelően távozásra szólította fel. Költözésünk időpontját lapzártakor még nem tudjuk, de a mi esetünkben nem kis hercehurcával járó művelet nyári tennivalóink sürgősségi listáján az elsők között szerepel.

*

Megújulási törekvéseinkhez igyekszünk felhasználni az olvasóinktól folyamatosan érkező ötleteket és véleményeket. Ilyen visszajelzések erősítettek meg bennünket például abban, hogy érdemes visszakanyarodni a Mikroszámítógép Magazin és az Alaplap hardveres hagyományaihoz, és nagyobb teret szentelni az ilyen témáknak. Kialakítottuk tehát a teszteket is tartalmazó középső hardverblokkot, amely iránt egyre nagyobb az érdeklődés, a mostani számban az oda tartozó cikkek és hirdetések megfelelelő elhelyezhetősége érdekében lapunk terjedelmét 4 oldallal meg is kellett növelni.

A jó elképzelések önmagukban azonban nem sokat érnek, ha nincsenek meg hozzá (vagy a körülmények folytán nem is teremthetők meg) a kivitelezés feltételei. Mindaz azonban, amibe most a 200. szám kapcsán a fent leírt módon belevágunk, megítélésünk szerint reális alapokon nyugszik.

Faklen Pál főszerkesztő
alaplap@mail.datanet.hu

Webszerver egy kártyán

Angliában, a Microsoft cambridge-i laboratóriumában a GSM mobiltelefonok intelligens kártyáin (smart card) működő webszervert fejlesztettek ki. A WebCamSim nevű miniszerver az MS Smart Card platformon alapul, és közönséges GSM telefonokba építve szöveges kiszolgálóként működhet. A kliens és a szerver közötti üzeneteket egy SMS átjáró (gateway) alakítja át az internetre csatlakoztatott számítógépek által is értelmezhető formátumúvá, maga a szerver pedig az MS Smart Card számára készült fejlesztői segédeszközzel programozható.

Kai Rannenberg, a kutatócsoport vezetője és a Microsoft Security Group tagja elmondta, hogy a szerver a GSM hálózatokban már meglévő kódolási eljárást és biztonsági rendszert használja, így olcsó és megbízható eszközként használható a hálózaton vásárolt áruk kifizetésére. A WebCamSim elméletileg arra is képes, hogy hagyományos, szöveges weblapokkal szolgálja ki a bejelentkező klienseket (a böngészőket), elsődlegesen azonban biztonságos „fizetőeszköznek" szánják. A telefontulajdonos a SIM-kártyán tárolt, jelszóval elérhető digitális kulcs segítségével hitelesítheti megrendelését, vagy erősítheti meg fizetési szándékát a hálózaton keresztül.

Mivel pénzről van szó, természetesen központi kérdés a biztonság, és mint minden esetben, itt is a leggyengébb láncszemet kell vizsgálni, amely a szakértők szerint az üzeneteket átalakító SMS átjáró. Nem szabad figyelmen kívül hagyni azt sem, hogy a GSM SIM-kártyák a kódolást csupán 40 bites kulccsal végzik. Az alkalmazott titkosító algoritmus állítólagos hibáját kihasználva 1999-ben az izraeli Weizmann Institute kutatóinak sikerült dekódolniuk GSM-üzeneteket. A jövőben bevezetésre kerülő harmadik generációs, UTMS hálózatok már jóval biztonságosabbak, 128 bites kulcsot fognak használni, viszont maguk a telefonok is bonyolultabbak lesznek, ami esetleg a biztonság rovására mehet. Az is tény, hogy jelenleg egy mobil készülék nem éppen ideális webszerver. A GSM telefonok memóriakapacitása nem túl nagy, az SMS-ek legfeljebb 160 karakterből állhatnak, azaz üzenetváltáskor a csomagok mérete erősen korlátozott, és minden különálló darabért fizetni kell. Mindazonáltal a szöveges üzenetek meglepően népszerűek Európában, a GSM Association 2000 decemberi elemzése szerint ebben az évben az emberek több mint 200 milliárd SMS-t fognak mobil telefonon egymásnak küldözgetni. *(ZDNet)*

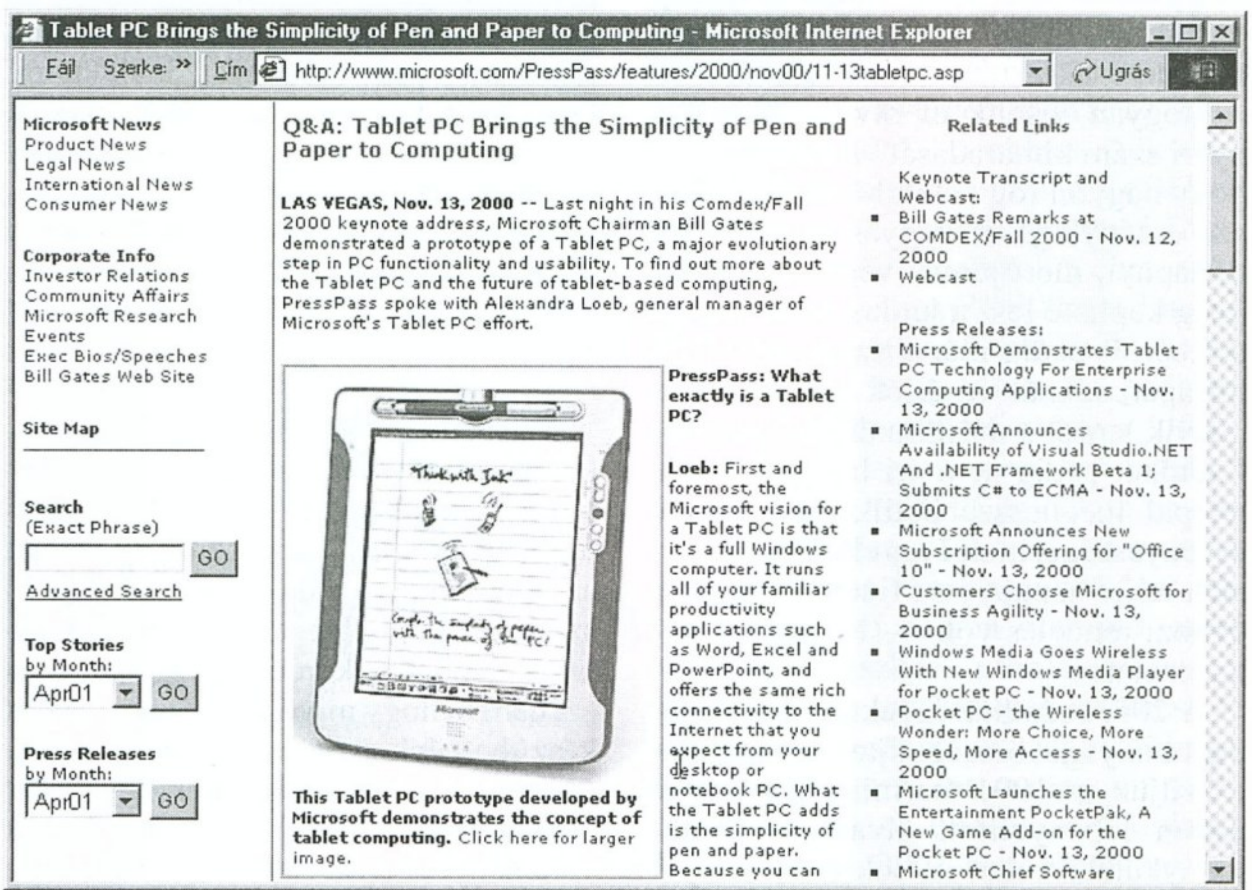
Whistler — béta 2

Március végén megjelent a Whistler Server operációs rendszer második béta-változata. Ebben a korábbi verziókhoz képest legtöbbet a Microsoft webkiszolgálója, a teljesen újratervezett Internet Information Server (IIS) változott. A béta 2 még nem tartalmazza a végleges Whistler Server minden funkcióját, amelyet egyébként a Windows 2000 Server, az Advanced Server és a Datacenter Server követőjének szánnak. A tervek szerint lesz harmadik béta is, amelyet kibocsátás előtti (release candidate, RC) változatok követnek az az év végére ígért piaci megjelenéséig. (A Whistler kódnevű operációs rendszernek a Windows 2000-hez hasonlóan több verziója lesz. A kliens végleges neve Windows XP, amelyből két változat készül: a Professional az üzleti szféra számára, a Home Edition a magánfelhasználóknak. A Whistler Servert kiszolgálónak szánják, arról azonban még nem született döntés, hogy milyen típusai lesznek, és az sem bizonyos, hogy ezek is XP néven kerülnek forgalomba. — A szerk.)

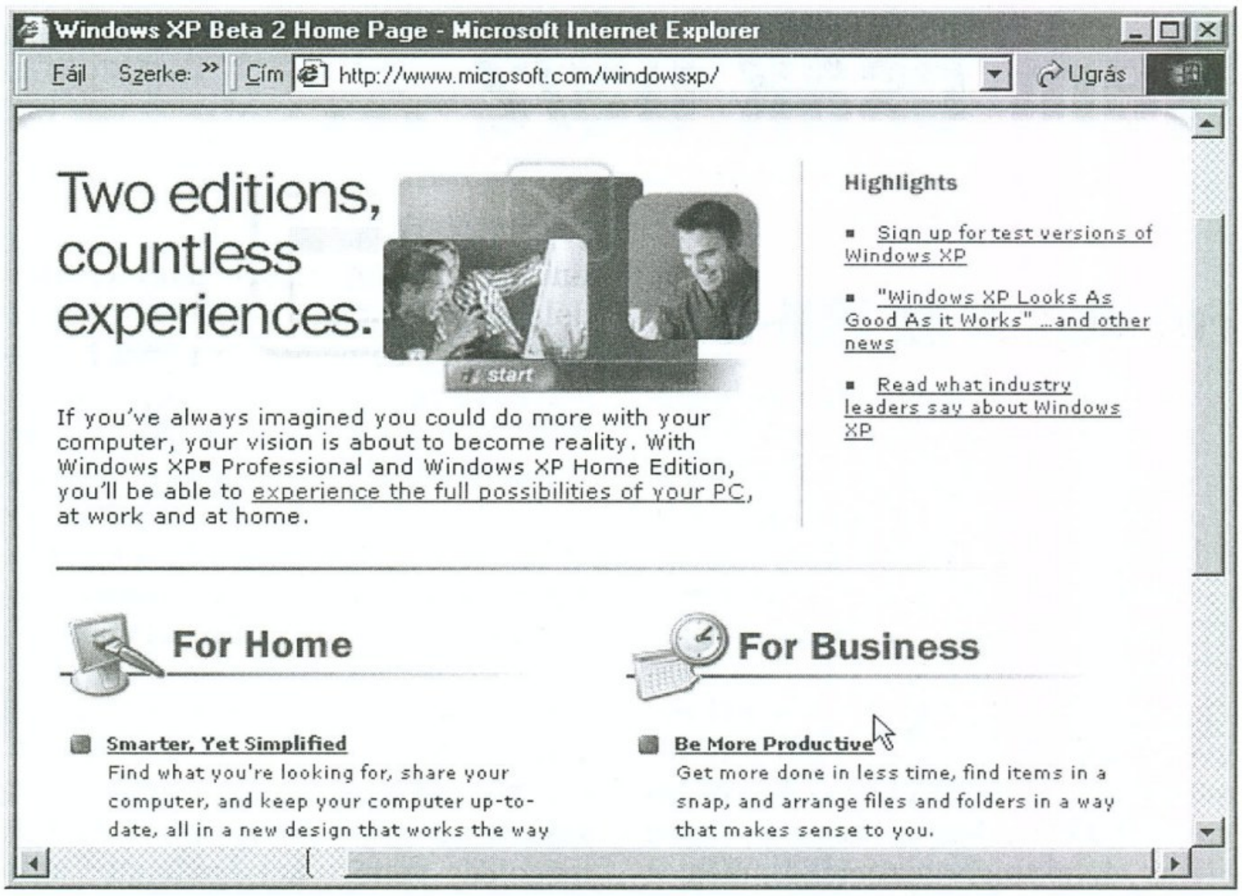
A már említett IIS 6.0-s változatának legfontosabb eleme a HTTP.SYS driver és cache. Ez a kernelen belül fog futni, és feladata a webről érkező kérések kezelése. Ha a keresett honlap megtalálható a gyorsítótárban, akkor átadja azt a kliensnek, ha nem találja, akkor továbbítja a kérést egy felhasználói módban futó másik IIS processznek. A gyorsítótáras kérések kiszolgálása így sokkal gyorsabb, mint az IIS 5.0 esetében, mert nincs szükség a kernel és a felhasználói mód váltogatására. Ez a felépítés egyébként a Tux nevű webes kiszolgáló architektúráját követi, ami nem is csoda, hiszen 2000-ben a Standard Performance Evaluation által végzett teljesítménymérések szerint az IIS 5.0 teljesítménye alaposan elmaradt riválisáétól. *(ZDNet – eWeek)*

Táblaszámítógép

Bill Gates bemutatta a weben való szörfölésre kifejlesztett számítógépét, a tablet PC-k microsoftos változatát, amely a Tablet PC nevet kapta. (A táblaszámítógép már elég régi elképzelés, melynek lényege, hogy az adatbevitel és vezérlés a táblaszerű képernyőn keresztül történik, például elektronikus tollal vagy érintéssel. Nevét a régi írótáblákra emlékeztető formájáról kapta. Több cég is próbálkozott már vele, de a várt eredményt még egyik sem tudta elérni. — A szerk.) A Microsoft a processzorokat gyártó Intellel és a Transmetával valamint öt PC-készítő vállalattal (közéjük tartozik a Toshiba, a Sony és a Compaq is) együttműködve dolgozik a Tablet PC kifejlesztésén, A legutóbbi Las Vegas-i Comdexen bemutatott mintadarab egy táskagép különálló képernyőjére emlékeztető, levélpapírnyi méretű és mintegy 5 centiméter vastagságú számítógép, vezeték nélküli interentes kapcsolattal és merevlemezzel felszerelve. Nincs viszont rajta billenytűzet. Az elektronikus toll



manapság már nem számít újdonságnak, ez a rendszer azonban fogadni tudja a normál kézírással történő adatbevitelt (úgy használhatjuk, mintha papírra jegyzetelnénk), és ötvözi azt a modern szövegszerkesztők lehetőségeivel. A kézzel írt szöveg a megszokott módon javítható a nem kívánt részek áthúzásával, a lap szélére írt jegyzetekkel és a megfelelő helyre történő nyilak bejelölésével. Mindezekből a Tablet PC egységes „gépelt" szöveget hoz létre, elvégezve a gépíró munkáját. A bemutatón a prototípus már képes volt kézzel írt betűk felnagyítására, kurziválására, kövérítésére, kivágására, mozgatására stb. A forgalomba hozatalt eredetileg 2001-re tervezték, de Bill Gates a bemutatón már 2002-ről beszélt. A táblaszámítógép operációs rendszere a várhatóan még idén megjelenő Windows XP lesz. A hardvert a Microsoft partnerei szállítják a Pocket PC-nél bevált konstrukció szerint, vagyis a Microsoft szállítja az operációs rendszert, a felhasználói felületet és az alkalmazásokat, továbbá ajánlásokat tesz a hardverre vonatkozóan, és mindezek alapján a többiek elkészíthetik saját márkájú táblaszámítógépüket. Az alapkonfiguráció tervezett paraméterei: 128 MB memória, 10 GB merevlemez, vezetéknélküli hálózati kommunikáció a 802.11 szabvány alapján, USB csatlakozó és egy tartóállvány. *(ZDNet – eWeek)*



Gyorsabb kernelfejlesztés

Több mint 50 programozó gyűlt össze az első olyan hivatalos találkozón, ahol a következő, 2.5-ös Linux kernel volt a téma. A VA Linux Systems egyik mérnöke szerint az a legfontosabb, hogy a fejlesztés ütemét felgyorsítsák, mert a 2.4-es (azaz 2.3-as) változaton még 1998-ban kezdtek el dolgozni, és csak 2001 januárjában fejezték be a munkát, ami túl hosszú idő. A 2.5-ös kernel tényleges fejlesztése még nem kezdődött el (ennek stabil változata a 2.6 vagy 3.0 sorszámot fogja kapni), erre még néhány hónapot várni kell, mert a fő cél egyelőre a stabilnak mondott 2.4-es rendszermag további „stabilizálása".

A rendezvényen azonban már körvonalazták a 2.5-ös fejlesztés fő irányvonalait. A Red Hat egyik képviselője a kernel és a tároló-rendszerek közötti interfész szerepét betöltő úgynevezett block device layer fejlesztésének fontosságát hangsúlyozta, ezáltal lehetőség nyílik majd 2 terabájtnál nagyobb kapacitású eszközök használatára. Szóba került továbbá a hálózati interfész fejlesztése, az aszinkron I/O műveletek támogatása és a virtuális memóriakezelés lehetőségeinek bővítése, többek között a vállalati nagy adatbázisok kezelésének megkönnyítésére. A Motorola előadója az SCTP (Stream Control Transmission Protocol) implementálásával kapcsolatos munkájáról számolt be, amely egy nemrégiben szabványosított átviteli réteg (transport layer) protokoll. Biztonsági kérdéseket is megvitattak a találkozón, a National Security Agency egyik alkalmazottja a hozzáférés szabályozásának lehetőségeit, valamint az NSA által fejlesztett Linux változatot mutatta be. Az új kernelben javulni fog továbbá a laptopok, az USB és PCMCIA csatlakozások támogatottsága, és egyebek közt szó volt új fájlrendszerek bevezetéséről és az SMP-ről is (symmetric multiprocessing). *(ZDNet)*

Galántai Zoltán – Mákos András

Megújuló Info

Május 8–12. között kerül sorra az Info 2001 nemzetközi informatikai és kommunikációtechnikai szakkiállítás a budapesti vásárvárosban. A Hungexpo a szakmai jelleg megerősítése érdekében fejlesztette tovább a rendezvényt. Az „A" pavilonban lévő szakkiállítás továbbra is olyan átfogó bemutató színehelye lesz, amely a számítástechnikai és távközlési szakemberekhez, az informatikai termékek széles felhasználói köréhez és valamennyi érdeklődőhöz szól. Ugyanakkor az első három napban (május 8–10.) életre hívott ICT-Fórum szűkebb szakmai körnek, az informatikai és gazdasági döntéshozóknak kínál kényelmes bemutatási és tárgyalási feltételeket a „B" pavilonban.

Április elejéig mintegy 150 cég jelezte részvételét az Infóra, 5400 m^2 nettó kiállítási területen, az ICT Fórumra pedig 25 kiállítót regisztráltak, mintegy 1100 m^2 területen. Az ICT-Fórum konferenciaprogramjának első napján kerül sor az országos informatikai tanulmányi versenyek eredményhirdetésére, valamint az „E" nélkül nem megy című rendezvényre, amelyen az e-business alkalmazásához szükséges IT követelményeiről, a mobil elektronikus pénztárcák biztonságáról, a digitális aláírás hazai bevezetésének megoldandó feladatairól hallhatnak előadásokat a résztvevők. Május 9-én „Behálózott gazdaság – tudástársadalom" címmel lesz egy másik konferencia, és szintén ezen a napon hangzanak el előadások az online és offline vevőkapcsolatokról (CRM). A május 10-i téma az e-book és az e-learning, és ekkor zajlik le a „Webmesterek utcája" rendezvény is.

Mindkét rendezvényen jelen lesz a Miniszterelnöki Hivatal informatikai kormánybiztossága. A nagyközönséggel megismertetik például a PC-programot és az Írisz-programot, az ICT-Fórumon pedig tájékoztatják a szakembereket többek között az információs társadalom kialakításának terveiről és eredményeiről, a digitális aláírás szabályozásának menetéről.

Nem csak OS/2 ...

Frissítések az IBM-től

Az IBM Software Choice weboldalán (http://www-4.ibm.com/software/os/warp/swchoice/) közzétett információk szerint a nagyvállalat a 2001-es év folyamán a következő, OS/2-vel kapcsolatos frissítések kibocsátását tervezi: TCP/IP 4.3.1, amely tartalmazni fogja a TCP/IP 4.3-hoz eddig kibocsátott javításokat, kezelni fogja a 2 K-nál nagyobb socketeket, és a jobb teljesítmény elérésének érdekében összehangolják a Java 1.3-as motorral is; logon kliens 4.4, amely az előző, még a Windows NT-hez kiadott 4.3 verzió Windows 2000-re átalakított változata; új, a Mozilla nyilvános (open-source) kódján alapuló OS/2-es böngésző, valamint az SSL kiegészítő modulokkal felvértezett, ugyancsak nyilvános kódon alapuló Apache webkiszolgáló.

OpenOffice

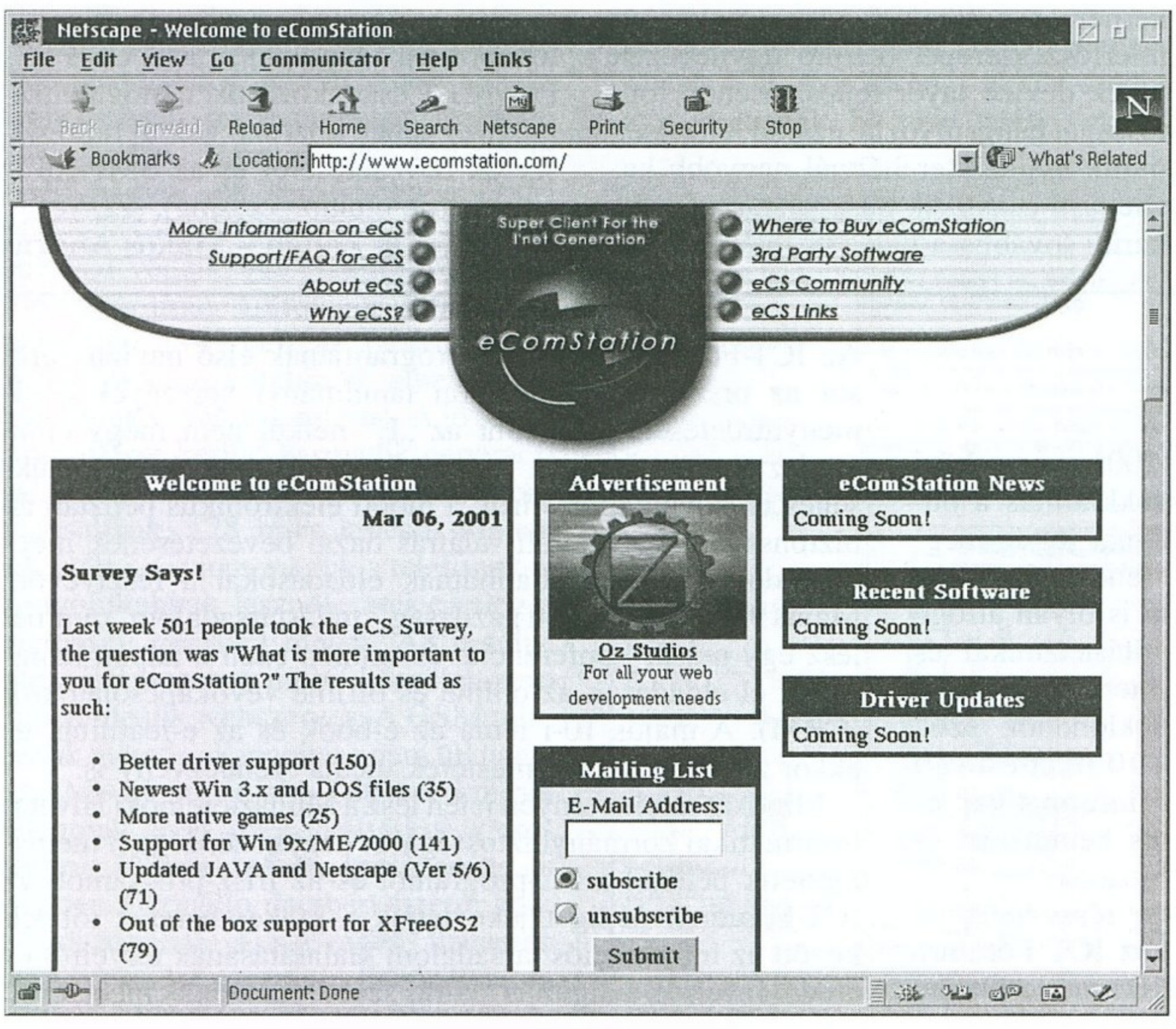
A StarOffice kódjának nyilvánossá tétele óta egyre több operációs rendszerre fordítják le ezt az ingyenes irodai alkalmazáscsomagot. A windowsos és linuxos változat mellett nagy erőkkel dolgoznak például a MacOS-en futó változaton, és nemrégiben bejelentették, hogy egy — egyelőre még csak két főből álló — német–amerikai csapat belekezdett az OS/2-es verzió elkészítésébe is. Más pletykák szerint orosz OS/2-es programozók is kacérkodnak a gondolattal. A feladat nem tűnik egyszerűnek, ugyanis a Sunnál annak idején nem sikerült lefordítani a StarDivision német cégtől megvásárolt 5.2-es verziót OS/2-re. Bár az Odin fejlesztőinek sikerült szóra bírniuk a Windowsra kiadott 5.2-es verziót, egy igazi OS/2-es megoldás ettől kétségtelenül jobb (gyorsabb, stabilabb) lenne. Az OpenOffice programról további információkat a http://www.openoffice.org oldalról lehet szerezni.

MultiDesk

Nemrégiben jelentette be Jürgen Dankoweit, hogy a Fixpak 13 (Warp 4.5) bevezetése után a termékében felmerült problémák miatt végleg felhagy a SecureOS/2 fejlesztésével, amely az OS/2 többfelhasználós alkalmazását tette volna lehetővé. Szerencsére nem sokat kellett várni a SecureOS/2 pótlására, mert a fejlesztés leállításának bejelentése után néhány héttel Cristiano Guadagnino kibocsátotta MultiDesk névre keresztelt termékét, amely gyakorlatilag ugyanazokat a funkciókat nyújtja, amellyel a SecureOS/2 is rendelkezett. Az egyelőre még csak bétaállapotban lévő alkalmazáshoz adott egyik segédprogram segítségével lehet a felhasználási helyeket létrehozni, személyre szabott munkaasztallal. Bizonyos fokig már a fájlhozzáférés korlátozása is megoldott, ám ezen a területen még sokat kell fejlődnie a programnak. A WPI fájlba csomagolt MultiDesk (md-0-1-9.wpi) a hobbes.nmsu.edu-ról tölthető le, kérdéseinket pedig a http://groups.yahoo.com/group/multidesk levelezési listán tehetjük fel.

EComStation honlap

Az eredeti tervekhez képest mégiscsak késik a január végére tervezett eComStation kiadása, s előreláthatólag csak március végén kapják majd készhez a vásárlók. A késés oka nem ismeretes, ám a yahoo.com-nál üzemeltetett ecomstation hírcsoportban megjelenő üzenetekből látszik, hogy a fejlesztők még mindig igen keményen dolgoznak a Warp 4.5 alapjaira épített operációs rendszer tökéletesítésén. Az utolsó pillanatban megegyeztek például az F/X Communicationsszel, és így beépíthetik az eCS-be a sokak által nagy megelégedéssel használt Injoy tárcsázóprogramot. A programozási feladatokon túl azonban egyre több figyelmet szentelnek a termék reklámozásának is. Megújították például a http://www.ecomstation.com címen található honlapot, amely igen tetszetős, mindenképpen érdemes ellátogatni oda azoknak, akik többet szeretnének megtudni az új OS/2-es kliensről.



UpdCD 1.4

Elkészült a Warp 4-es telepítő CD-k frissítésére használható UpdCD program 1.4-es kiadása. A legfontosabb újítások között lehet említeni a népszerű faxprogram, a PMFax Lite (FaxWorks) frissítésének, valamint az Odin és az XWorkPlace támogatását, illetve a Warp 4 programadatbázisának (database.txt) frissítését. Az ezzel a verzióval készített CD-ről már közvetlenül is lehet bootolni és telepíteni, nincs szükség többé a lassú és sérülékeny floppykra. Az UpdCD 1.4 letölthető a http://xenia.sote.hu/~kadzsol/rexx/sajat/updcd.htm oldalról. A használat során fel-

merülő problémákkal a http://groups. yahoo.com/group/updcd levelezési listához érdemes fordulni.

Tonigy Audio CD IFS

Ismét új telepíthető fájlrendszerrel gazdagodott az OS/2. A furcsa hangzású Tonigy-re keresztelt Audio CD IFS-szel zenei CD-k sávjait lehet WAV vagy RAW fájlként elérni. Ennek azért van nagy jelentősége, mert így közvetlenül, mindenféle előzetes konverzió nélkül lehet a CD-n lévő adatokhoz hozzáférni. Ha például a munkaasztalon keresztül tekintjük meg egy zenei CD tartalmát a Tonigy fájlrendszer telepítése után, akkor a CD nem elérhető üzenetei helyett WAV fájlokat fogunk látni, amelyeket dupla egérkattintással akár azonnal le is játszhatunk. A shareware alkalmazás a http://tonigy.hypermart.net oldalról szerezhető be.

NetDrive 1.0.7

A Blueprint Software Works jóvoltából egyre-másra jelennek meg a NetDrive fájlrendszer újabb verziói, amelyekkel a lokális vagy a távoli (például egy ftp-kiszolgálón elhelyezkedő) könyvtárakhoz lehet meghajtóbetűket rendelni, hogy azokat azután közönséges meghajtóként lehessen kezelni. A cikk írásának pillanatában a legújabb, 1.0.7-es verzió lényeges újítása a tökéletesített kezelőfelület (NetDrive Control Panel), amellyel a virtuális könyvtárak tulajdonságait lehet menet közben beállítani. Az egy hónapig ingyenesen kipróbálgatható shareware a http://www.blueprintsoftware.com/netdrive könyvtárban található meg.

```
C:\WINNT\System32\cmd.exe
D:\TTT>winwpi
WinWpi - WarpIN archive viewer / unpacker for Windows v0.1
(c) 2001 Cornelis Bockemühl
WarpIN backend (archiver): (c) 1998-2000 Jens Bäckman, Ulrich Möller
BzLib2: (c) 1996-2000 Julian R Seward
Published under the GNU General Public Licence (see the COPYING file).

Use: winwpi <archive> -<cmd> [<package>] [<files>]
With: <archive>: name of the archive file
      -<cmd>:    -l = list files
                 -p = list packages
                 -x = extract files
                 -X = extract script
      <package>: optional package number
      <files>:   file name or mask (using * and ? jokers)

D:\TTT>winwpi xwp.wpi -p
Application:            Test application
Revision:               1
Author:                 Ulrich Möller
Internet download:      N/A
Operating system:       1
Number of packages:     3
Min. WarpIN revision:   3

Nr   Files   Name
------------------------------------------------------------
 1      60   Pck001
 2      10   Pck002
 3       4   Pck003

D:\TTT>_
```

WPI-nézegető

Még csak néhány hónap telt el a WarpIn telepítő WPI fájlformátumának bevezetése óta, máris gomba módra szaporodnak az ebben a kiszerelésben terjesztett OS/2-es alkalmazások. Néhány hete a legismertebb OS/2-es ftp-kiszolgáló, a hobbes.nmsu.edu kezelője is bejelentette, hogy a ZIP formátum mellett ezentúl támogatni fogják a WPI fájlformátumot is. A széles körű elterjedésnek persze nem biztos, hogy mindenki örül. A WPI fájlok ugyanis eddig csak a WarpIn telepítő és az OS/2 alatt futó WarpZip program számára voltak értelmezhetők, s ha valaki mondjuk Windows alatt akart belenézni egy WPI csomagba, annak nem volt szerencséje. Most már erre is van megoldás, ugyanis megjelent a WinWpi program 0.1-es verziója, amellyel Windows alól is kezelhetővé váltak a WPI fájlok. Igaz, hogy a program még csak szöveges felülettel rendelkezik, ám ha valaki nagyon megszorul, az is jobb a semminél.

A winwpi.zip fájlban lévő alkalmazás letölthető a http://hobbes.nmsu.edu webhelyről.

Kádár Zsolt
kadzsol@xenia.sote.hu

Adathordozók őrségváltása I.

A CD-író szerves tartozékká válik

A CD-írók árának folyamatos esése egyre kelendőbbé, egyre több ember számára elérhetővé teszi ezeket a termékeket. Egyidejűleg az egyszer írható korongok ára is olyan drasztikusan csökkent, hogy már majdnem azonosak a 3,5 collos floppy árával. Egy kis túlzással azt mondhatjuk, hogy adattárolásban a CD a merevlemez vetélytársa lett, bár jóval lassúbb annál. Mobil adathordozóként a CD viszont eddig talán csak azért nem tudta felváltani a floppyt, mert nincs minden PC-ben CD-író. Előbb-utóbb azonban lesz.

Kezdjük a CD-írás megközelítését egy kis számolással. Bár az újraírható CD-k között vannak olcsóbb és drágább változatok, ha az egyszerűség kedvéért darabját 650 forintosnak vesszük, akkor egy megabájt 1 forintba kerül. A 20 gigabájtos merevlemez ára is nagyjából 20 ezer forint körül van, így mindkettő fajlagos költsége azonos: 1 MB = 1 forint. Ennél a számolásnál azonban a CD-t egy kicsit drágítottuk, a merevlemezt pedig a valóságosnál olcsóbbnak tüntettük fel, ugyanakkor kihagytuk a CD-író árát. Ha ezeket a tételeket is figyelembe vesszük, és mindent reális áron számolunk, akkor ma összességében mégiscsak hasonló költségekkel jár az adattárolásnak ez a két formája.

Mindkét tárolási módnak vannak előnyei és hátrányai. A winchester gyors, de többnyire helyhez kötött. A CD hordozható, de az adatok mozgatása jóval több időt igényel. Ha mobil rekeszt használunk a merevlemezhez, akkor nem vagyunk helyhez kötve, de sokan mégis szívesebben hordanak maguknál inkább egy 3–400 forintos korongot. Floppy helyett is egyre gyakrabban CD-re írnak fel mindent, néha még akkor is, ha az csak 2–3 megabájt. Valószínűleg az újraírható CD-RW fogja felváltani az 1,44 MB-os floppyt, nagyobb adatmennyiségek fuvarozására pedig talán a mobil merevlemez lesz a legjobb. A helyzet persze nem ennyire egyszerű. Már régóta beharangozták a DVD-írókat és a DVD-R lemezeket, amelyek 10-szer akkora kapacitást kínálnak, mint a CD-k. Egyelőre azonban ezek ára igen magas, és még meg kell oldani egy-két kompatibilitási problémát. (Az Új Alaplap következő havi tesztje a DVD-vel foglalkozik.)

CD-szabványok

A Sony és a Philips már a 70-es években elkezdett közösen fejleszteni egy optikai adattároláson alapuló eszközt, de az első szabvány (a könyv borítója alapján elnevezett Vörös Könyv, a Red Book) csak 1982-ben született meg. A későbbi szabványok (és más színű könyvek) mindegyike ehhez nyúlik vissza, ez rögzítette a legelső CD-formátumot, az audio-CD-t (CD-DA, compact disc digital audio), a régi, hagyományos zenei CD-k formátumát. Azért használjuk a „régi" jelzőt, mert a ma megjelenő zenei CD-k jelentős hányadát már ellátják mindenféle multimédiás képességekkel, és ezeknek a lemezeknek a formátuma sem CD-DA.

Az audio-CD hangsávokra (track) van osztva (maximum 99-re), amelyek között 2 másodperc szünet van. A lemez első két másodpercét a track-ek listája, a tartalomjegyzék (TOC, table of contents) foglalja el.

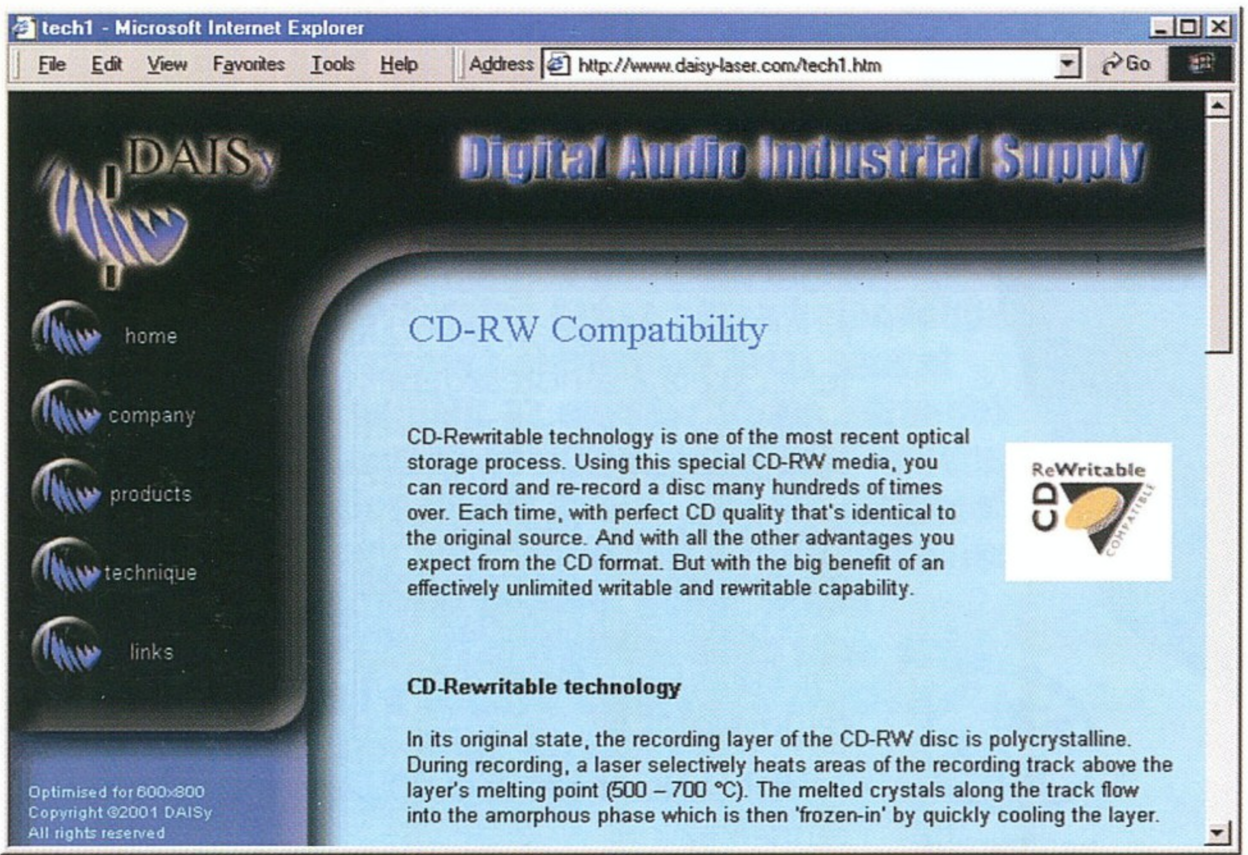
Az 1984-ben megjelent Sárga Könyv (Yellow Book) biztosította a szektorok felismerését, és további hibajavító bitek definiálásával tette biztonságosabbá az adattárolást. A hagyományos számítógépes adatok tárolására alkalmas CD-ROM (read only memory) szabványt már ebben a kötetben definiálták. A Sárga Könyvre épül a CD-ROM XA szabvány (Yellow Book eXtended Architecture), amely lehetővé teszi, hogy a hanganyagot, a képeket és a hagyományos számítógépes adatokat tartalmazó szektorok vegyesen helyezkedjenek el. A Sárga Könyv szabványa szerint felkészített meghajtók megfelelő driver segítségével képesek olvasni ezt a formátumot is.

A következő kiadás, a Zöld Könyv határozza meg a CD-I (interactive) formátumot, amelyet elsősorban multimédiás alkalmazásokhoz készítettek szintén a Sony és a Philips mérnökei. Ez a specifikáció azért is érdekes, mert nemcsak a lemez tartalmára vonatkozóan ad útmutatást, hanem meghatározza az egész hardver- és szoftverkörnyezetet, a CD-n tárolt audio, video és szöveges adatok kapcsolatait is. A CD-I formátumú lemez olvasására a régebbi CD-ROM meghajtók nem alkalmasak, mert a CD-I lemezen nincsen TOC.

A Narancssárga Könyv (Orange Book) második publikációja mérföldkőnek számít a CD fejlődésében. Ez az 1992-ben megjelent kötet definiálja a CD-R (CD Recordable, azaz írható CD) szabványt és a multisession formátumot. Előzőleg a CD gyártása kis példányszámban nagyon költséges volt, a lemezeket a gyári mintalemez (master) segítségével, sajtolással állították elő, és a lemez egyik rétege alumíniumból készült. A Narancssárga Könyv specifikációja szerint az alumínium réteget felváltották aranyréteggel, alá pedig fényérzékeny réteget helyeztek el, amelyből nagy energiájú lézersugár segítségével égetik ki a felesleges részeket, így az előállítási költség jelentősen csökkenthető.

A session (magyarul legjobban a szakasz elnevezés közelíti meg, ezért ezt használjuk rá) a CD-n tárolt adatok egyszerre felírt nagyobb blokkját jelenti. A multisession (többszakaszos) eljárás lényege pedig az, hogy a tulajdonképpen „egyszer írható" CD-R lemezekre több részletben egymás után mindaddig hozzá lehet írni valamit, amíg a lemez be nem telik. Mindössze az egyes felírások után nem szabad lezárni a lemezt. Ez a lehetőség egy ügyes trükknek köszönhető: minden új szakasz elejére felírnak egy tartalomjegyzéket (TOC), amely az előzőleg már felírt adatok tartalomjegyzékét is tartalmazza. A többszakaszos felírású CD-ket kezelni képes meghajtók megkeresik a legutolsó tartalomjegyzéket, és annak alapján tájékozódnak.

A Narancssárga Könyv 1994-ben kiadott, harmadik része specifikálja a CD-RW (ReWritable, újraírható) szabványt. A legfontosabb kérdés ismét a



kompatibilitási problémák megoldása volt, ami azonban nem sikerült maradéktalanul, így a régebbi meghajtók nem képesek olvasni a CD-RW lemezeket.

További CD-formátumok

— CD-G (Graphics). Az audio CD kiegészítése, segítségével a lemez lejátszása közben egyszerű grafikákat is be lehet tölteni, bár elég lassan.

— CD-Text. Lehetővé teszi az audio lemezeken a szöveges információk (dalszövegek, dalcímek stb.) tárolását.

— Mixed Mode CD. Adatokat és hangsávokat is tartalmaz. Először az adatok vannak rögzítve, amit egy vagy több audiosáv (track) követhet. (Ezzel csak olyankor van gond, ha például betesszük egy régi CD-lejátszóba, amely gyanútlanul „lejátssza" az első szakaszba bemásolt Win95-öt...)

— CD-Extra. Ugyanúgy adatokat és hangsávokat tartalmaz, mint a Mixed Mode CD, azzal a különbséggel, hogy míg az előzőben minden egy szakaszban (session) helyezkedik el, ez már többszakaszos (multisession) formátumú. Az elsőben audiosávok vannak, és a CD-lejátszó csak ezeket látja, a többi szakaszban viszont azt tesszük, amit akarunk.

— Video CD. MPEG1 kódolású, elég jó minőségű filmrögzítési mód.

Mint látható, a CD-k családfáján rengeteg ág található, és ez okozza a legtöbb kompatibilitási problémát. Sokszor előfordul, hogy egy régi meghajtó nem képes olvasni a nagy sebességgel felírt CD-R vagy CD-RW korongokat. A CD-család túlburjánzása pedig ma is folytatódik, egyre-másra állnak elő a gyártók az új szabványokkal. Az idei trónkövetelő a DDCD (double density CD), azaz kétszeres sűrűségű CD (kisebb a pitméret és a sávok közötti sortávolság), így 1,3 GB adat tárolása válik lehetővé — mondanunk sem kell, hogy megintcsak a régi meghajtókkal való inkompatibilitás árán.

Sebességkorlát

A piacon rövid idő alatt lényeges szerkezeti változás megy végbe. A csak olvasásra alkalmas CD-meghajtók fokozatosan háttérbe szorulnak, mert a CD-RW és a DVD egységek ára egyre kedvezőbb, és mindegyik ellátja a szimpla olvasó feladatát is. Néhány évvel ezelőtt a gyártók még a leggyorsabban olvasó egységek előállításában versengtek. Ez a licit azonban „kifulladt", mert a fordulatszám egyszerű fizikai korlátok miatt nem növelhető bizonyos határon túl. Kis szögsebességek mellett a CD-lemez viszonylag egyenletesen forog, de nagyobb fordulatszám esetén akár „robbanásveszélyt" is jelenthet a korong egyenetlen tömegeloszlása, például egy ráragasztott matrica miatt. Ilyenkor a lemez forgása instabillá válik, precessziós (körpályától eltolódó) mozgás lép fel, az adatolvasási képesség drasztikusan lecsökken, esetleg az olvasófej is tönkremegy. A CD-olvasás sebességét ezért már csak új technológiákkal lehet majd növelni, például több olvasófej alkalmazásával.

Olvasási és írási technika

Többfajta CD-olvasási és CD-írási technika terjedt el, tekintsük át röviden ezeket. A CD-írás a legtöbb esetben konstans adatátviteli sebesség mellett történik. A CD-n az adatok sűrűsége állandó, ellentétben például a régi bakelitlemezekkel. A régi lemezjátszókon a motor állandó szögsebességgel forgatta a lemezt, függetlenül az olvasófej helyzetétől. Mivel az egyenletes lejátszáshoz egységnyi idő alatt ugyanannyi adatnak kellett áthaladnia az olvasófej alatt, ezért a külső sávokon az adatsűrűség kisebb volt, mint a belső sávokon. A CD esetében ellentétes a helyzet, itt az adatsűrűség konstans, és a CD forgatásának sebessége változó. Állandó szögsebesség mellett a CD belső sávjainak kerületi sebessége kisebb lenne, mint a külső sávoké, így a külső részen azonos idő alatt több adat haladna át az olvasófej alatt, mint belül, és az adatolvasási sebesség bentről kifelé nőne. Mivel állandó adatolvasási sebességre van szükség, a szögsebességet a bentről kifelé való íráskor állandóan csökkenteni kell. A módszer hátránya, hogy a belső sávok maximális adatátviteli sebessége korlátozza a külső kerületek átviteli lehetőségét is. Ennek a CD-írási technikának a neve a CLV (constant linear velocity, állandó kerületi sebesség).

A másik, ennél fejlettebb technika a CAV (constant angular velocity, állandó szögsebesség). Mint a neve is elárulja, itt nem az adatátvitel lesz állandó, hanem a szögsebesség. Ez a módszer jobban alkalmazkodik a meghajtó fizikai paramétereihez, ezért nagyobb adatmozgatási sebességet lehet elérni.

A harmadik megoldás, a részleges CAV (P-CAV, Partial-CAV) az előző két módszer vegyítése. A belső területek írásakor a meghajtó konstans szögsebességgel pörgeti a lemezt (CAV), ennek következménye, hogy a kezdeti átviteli sebesség bentről kifelé nő. Amikor azonban a sebesség elér bizonyos küszöböt, akkor a meghajtó átvált állandó átvitelre. A tesztnek a táblázatban mért részleges CAV értékeit illetően meg kell említeni, hogy a meghajtó bizonytalan olvasás esetén is alacsonyabb sebességre válthat. Miután több lemezzel próbálkozva ez mindig máshol következett be, ilyen optikai hibákkal szembeni érzékenységet lehet feltételezni, vagyis csak látszólagos parcialitásról van szó, nem valóságosról.

A Z-CLV (Zoned CLV, zónánként állandó kerületi sebesség) a P-CAV módszerrel ellentétben először állandó kerületi sebességgel (CLV) működik, majd állandó szögsebességre (CAV) vált, végül CLV-ként fejezi be a műveletet.

Mérési körülmények

Az IDE-s eszközöket minden esetben a számítógép secondary slave csatornájára kötöttük, és a Windowsban enge-

délyeztük a DMA hozzáférést. Az író szoftverek a következőek voltak: Nero 5.0, Easy CD Creator 4.01 és WinOnCD 3.7. Mindig az adott meghajtónak legjobban megfelelő programmal írtunk.

A tesztelést a népszerű CDSpeed99 legfrissebb verziójával végeztük. A CD-ROM olvasási mérésekhez az Új Alaplap 2001. márciusi számának CD-mellékletét használtuk, a CD-R olvasási mérések alapjául egy 16-szoros sebességgel írt Dysan lemez szolgált. A CD-RW mérésekhez egy 10x-es Sony HighSpeed CD-RW korongot nyúztunk. Minden íráskor az Új Alaplap márciusi CD-jének tartalmát rögzítettük az adott médiára, grabbeléskor ugyanazokat a korongokat alkalmaztuk. A táblázat a CDSpeed99 tesztek eredményét, a gyári adatokat és az önálló mérési eredményeket tartalmazza. A táblázatban szereplő legfontosabb mérési szempontjaink a következők voltak.

- Grabbelés

Az audio CD zenei adatának .wav fájlformátumba való leképezése, ehhez a népszerű Audiograbber programot használtuk, amely hibajavítás nélkül végigolvassa az audio CD-t. Ez viszonylag nagy sebességet eredményez, de gyengébb CD-olvasók esetében a minőség rovására mehet, azaz a leképzett hang kattogni fog.

- Átviteli sebesség

Az adatoknak adott időegység (1 másodperc) alatt egyik helyről a másikra való mozgatási sebessége. Ebből mértünk írási és olvasási sebességet is. Az olvasási sebesség függ attól, hogy a CD mely területéről, mekkora és milyen fájlokat olvas a meghajtó. A külső sávokon a kerületi sebesség nagyobb (CAV esetén), azaz adott idő alatt több adatot lát az olvasófej. Tehát az olvasási sebesség bentről kifelé nő (ezért szerepel a táblázatban átlagos, eleje és vége bejegyzés). Az írási sebesség mérésekor részben a CDSpeed-99 eredményeire támaszkodtunk. Ahol nem szerepel adat, ott a program nem tudta elvégezni a mérést.

- Maximális írási sebesség

Arra voltunk kíváncsiak, hogy a specifikációban meghatározott legnagyobb szorzójú íráskor mekkora valódi adatátviteli sebességgel dolgozik a meghajtó. A CD-DA írási sebességet külön teszteltük, itt arra voltunk kíváncsiak, hogy a maximális sebességgel felírt audiolemez élvezhető minőségben visszaolvasható-e az asztali lejátszóval. Ezen a ponton gyakorlatilag mindegyik meghajtó megbukott. Azt tanácsoljuk tehát, hogy ne írjanak 2x-esnél vagy 4x-esnél gyorsabban audio CD-t, mert sem az asztali lejátszók, sem a számítógépes meghajtók nem képesek hiba nélkül visszajátszani a nagyobb sebességgel felírt lemezeket.

- Átlagos elérési idő

Az adatok megtalálásának ideje. A tesztprogrammal három értéket tudtunk mérni. A „véletlen" bejegyzés a lemez teljes felületén végzett véletlenszerű elérési parancsok végrehajtási idejét mutatja, az „1/3" a fejáthúzáshoz szükséges idő a lemez 1/3-án, a „teljes" pedig a teljes felületén.

- CPU-terhelés

A CD-ROM adott sebességű olvasása esetén a processzor terheltségének százalékosan megadott értéke.

- Lemezpörgetési idő

A lemez leállításához és felpörgéséhez szükséges idő.

- Tálcakiadási és betöltési idő

A lemezkiadó gomb megnyomásától vagy a kilökési (eject) parancs kiadásától a tálca megmozdításáig eltelt idő, a betöltési idő ennek a műveletnek a fordítottjára vonatkozik.

- Lemezfelismerési idő

A lemez tartalomjegyzékének (TOC) beolvasásához szükséges idő, ez CD-formátumonként jelentősen különböző lehet.

Általános tapasztalatok

A modern meghajtók nagy része támogatja a Narancssárga Könyv (2. és 3. része) által deklarált írásmódokat: Track at Once, Multisession, Disk at Once (DAO), Session at Once, Packet Writing. Gyakorlatilag az összes meghajtó képes volt kezelni az elterjedtebb CD-formátumokat, a kényes pontot a CD-Text írása jelentette (a táblázatban feltüntettük ennek meglétét).

Mindegyik egység rendelkezik az audiolemez lejátszásához szükséges digitális és analóg audiokimenettel, ezt a táblázatban külön nem tüntettük fel. Sajnálatunkra azonban egyik meghajtón sem volt Play gomb, amelynek segítségével zenei CD-t tudtunk volna lejátszani szoftver közbeiktatása nélkül.

- Freecom

A Freecom nem maga készíti a meghajtókat, hanem más gyártók termékeit dobozolja be. Elsősorban a versenyképes árra és az igényes körítésre helyezik a hangsúlyt. A tesztben szereplő belső

CD fájlrendszerek

ISO 9660

Az International Standards Organization által 1984-ben elfogadott szabvány. A fő irányelv az addigi inkompatibilitás helyett mindenki által olvasható fájlformátum létrehozása volt. Manapság is kiterjedten alkalmazzák, népszerűségét annak köszönheti, hogy a legtöbb operációs rendszer kezelni tudja. Ennek persze ára van: mind a kialakítható könyvtárstruktúra mélysége, mind a fájlok és a könyvtárak elnevezése korlátok közé van szorítva. A könyvtárszerkezet maximum 8 szintű lehet (főkönyvtár + 7 alkönyvtár), a fájlneveknek a 8+3 (név + kiterjesztés) forma szabályaihoz kell igazodni, a könyvtárnevek pedig maximum 8 karakterből állhatnak. Sajnos az eredeti tervekkel ellentétben ez a szabvány az operációs rendszerek közül csak a DOS számára volt optimális.

Joliet

Az ISO 9660 szabvány kiterjesztése. Azért volt rá szükség, mert az operációs rendszerek kinőtték a régi szabvány kereteit (leginkább a Windows 9x-ekől van szó). Sajnos az új szabvány inkompatibilitást teremtett a régi rendszerekkel. A hosszúfájlneves (maximum 64 karakteres) elnevezést és a könyvtárstruktúra szerkezetét a Microsoft igényei határozták meg.

UDF

A Universal Disk Format az Optical Storage Technology Association által CD-R és CD-RW lemezekhez kidolgozott szabvány, az ISO 9960 továbbfejlesztése. Az íróprogram által megformázott, UDF formátumú lemezre (kb. 550 MB) bármilyen fájlkiírási műveletet végrehajtó programmal írhatunk, tehát például Wordből is közvetlenül elmenthetjük az anyagot a CD-re. A korongra csomagokban kerülnek fel az adatok (packet writing). A CD-R lemezekre addig dobálhatjuk rá a csomagokat, amíg az meg nem telik. A CD-RW lemezeken lehetséges a törlés is, ilyenkor a CD-író programok (Direct CD, PacketCD) szabadítják fel a lemeznek újraírásra alkalmassá váló területét.

GYÁRTÓ	TDK	Yamaha	Teac		Sony		Freecom		
TÍPUS	CyClone	CRW 2100E	CD-W 58E	CD-W 512E	CRX 0811	CRX 160E	8/4/32	8/8/32	12/10/32
Csatolás	IDE	IDE	IDE	IDE	IDE	IDE	IDE	IDE	IDE
Névleges adatátvitel (írás/újraírás/olvasás)	16/10/40	16/10/40	8/8/32	12/10/32	8/4/32	12/8/32	8/4/32	8/8/32	12/10/32
Firmware verziószám	5.29	1.0G	1.0A	2.0A	MYS2	1.0E	2.0	1.0A	2.0A
Pufferméret	2 MB	8 MB	2 MB	4 MB	2 MB	4 MB	2 MB	2 MB	4 MB
CD-Text írási mód	+	+	–	–	+	+	–	–	–
Pufferkiürülés elleni védelem	BURN-Proof	Nincs	Nincs	BURN-Proof	Nincs	Nincs	Nincs	Nincs	BURN-Proof
Túlírás támogatása	+	+	+	+	n.a.	n.a.	–	+	+
Olvasási mód, CD-ROM	CAV	CAV	CAV	CAV	CAV	CAV	P-CAV	CAV	CAV
Olvasási mód, CD-R	CAV	CAV	CAV	CAV	CAV	CAV	P-CAV	CAV	CAV
Olvasási mód, CD-RW	CAV	CAV	CAV	P-CAV	P-CAV	P-CAV	P-CAV	CAV	P-CAV
Írási mód, CD-R	CLV	P-CAV	CLV	CLV	n.a	P-CAV	n.a.	CLV	CLV
Írási mód, CD-RW	CLV	CLV	CLV	CLV	n.a	CLV	n.a.	CLV	CLV
Mérési eredmények									
Gyári audio CD grabbelés	23,60	23,40	12,30	23,63	12,80	12,30	8,00	12,30	23,63
Írt audio CD grabbelés	19,30	23,00	12,30	18,70	12,00	11,60	7,30	12,30	18,70
Max CD-R írás	13,90	14,50	7,30	10,10	7,50	10,01	7,30	7,30	10,10
Max CD-DA írás	13,83	12,70	4,03	10,40	7,60	9,70	3,40	7,35	10,40
Max CD-RW írás	9,30	7,10	7,50	9,40	3,70	7,50	3,60	7,50	9,40
CDSpeed99 és DVDSpeed99 teszteredmények									
Adatátviteli sebesség									
Olvasás, átlagos, CD-ROM	30,25	28,70	23,94	24,34	24,07	24,19	24,81	23,94	24,34
Olvasás lemez eleje, CD-ROM	18,22	16,95	14,00	14,58	14,22	14,49	14,93	14,00	14,58
Olvasás lemez vége, CD-ROM	38,79	37,86	31,00	32,00	31,45	31,81	31,12	31,00	32,00
Olvasás, átlagos, CD-R	30,26	28,68	24,11	24,45	24,05	24,27	24,80	24,11	24,45
Olvasás lemez eleje, CD-R	18,11	16,84	14,35	14,56	14,13	14,25	14,85	14,35	14,56
Olvasás lemez vége, CD-R	39,00	37,89	31,74	31,96	31,84	32,06	31,10	31,74	31,96
Olvasás lemez átlagos, CD-RW	25,18	29,08	23,87	22,14	14,81	15,50	25,09	24,11	22,14
Olvasás lemez eleje, CD-RW	14,97	17,09	13,98	14,59	8,61	9,07	15,00	14,25	14,59
Olvasás lemez vége, CD-RW	33,03	38,44	31,01	24,2	19,56	20,49	31,07	30,94	24,2
Írás lemez átlagos, CD-R	8,00*	15,60	8,00	12,00	n.a.	11,99	n.a.	8,00	12,00
Írás lemez eleje, CD-R	8,00*	12,15	8,00	12,00	n.a.	11,91	n.a.	8,00	12,00
Írás lemez vége, CD-R	8,00*	16,02	8,00	12,00	n.a.	11,98	n.a.	8,00	12,00
Írás lemez átlagos, CD-RW	10,00	8,00*	8,29	10,00	n.a.	7,99	n.a.	8,29	10,00
Írás lemez eleje, CD-RW	9,97	8,00*	8,29	10,00	n.a.	7,90	n.a.	8,29	10,00
Írás lemez vége, CD-RW	9,88	8,00*	8,29	10,00	n.a.	8,02	n.a.	8,29	10,00
Elérési idő									
Véletlen, CD-ROM (msec)	131	134	77	86	125	114	86	77	86
1\3, CD-ROM (msec)	139	153	84	89	138	119	95	84	89
Teljes, CD-ROM (msec)	250	224	157	164	253	151	163	157	164
Véletlen, CD-R (msec)	121	133	77	81	126	121	89	77	81
Véletlen, CD-RW (msec)	126	136	97	132	133	145	90	97	132
CPU terhelés 1x/2x/4x/8x olvasás esetén (%, CD-ROM)	1/3/6/11	1/3/13/9	1/3/5/10	1/3/5/16	2/5/10/19	1/2/4/7	2/4/7/14	1/3/5/10	1/3/5/16
Lemezfelpörgési/leállási idő (sec)	1,41/5,17	3,87/3,94	1,18/2,58	3,47/6,09	1,44/4,41	2,68/3,59	3,16/3,59	1,18/2,58	3,47/6,09
Tálcakiadási idő (sec)	3,32	1,58	1,64	2,34	1,32	1,73	0,74	1,64	2,34
Lemezbetöltési idő (sec)	8,15	1,69	7,93	10,62	2,84	7,20	0,02	7,93	10,62
Lemezfelismerési idő (sec, CD-ROM)	0,73	5,66	0,02	0,02	7,54	0,03	6,42	0,02	0,02
Irányár vagy ajánlott kiskereskedemi ár (Ft)	79.200	68.000	28.000	42.500	32.890	61.490	32.000	35.000	48.000
Tesztpéldány	Conflex	Storage System	Komel		Enternet 2001		Napfény		

* A mérőprogram által nem detektált érték.

Freecom írók majdnem mindegyike azonos kiegészítőkkel van ellátva, csak a 12/10/32-es modellhez nem adnak újraírható CD-t. Mindegyik meghajtóhoz ugyanolyan segédkönyvet mellékeltek, amely az adott meghajtóról szóló részletes specifikációi helyett csak általános elveket és módszereket tartalmaz.

A 8x/4x/32x-es meghajtó tálcamegoldása eltér az általánostól, a leméztartó egy kis kapu mögül bújik elő, és érezhetően gyorsabban reagál az „eject" parancsra, mint a többi tesztkészülék. A 8x/8x/32x-es Freecom Bulk fantázianevű író valójában egy TEAC W58E, a 12x/10x32x-es író pedig egy TEAC W512E(B) meghajtót rejt magában, ezért nem véletlen, hogy a táblázat megfelelő oszlopai megegyeznek.

- Sony

Három Sony meghajtó járt tesztlaborunkban, ebből kettő a CRX160E (12/8/32) típus dobozos és OEM verziója, a harmadik a CRX0811 (8/4/32) modell. A szoftverellátottságra nem lehet panasz, az egyetlen dolog, amit kifogásoltunk, hogy a dokumentáció csak a szoftver CD-jén volt rajta. Az OEM és a dobozos CRX160E meghajtók teljesítménye azonos volt.

A kisebb Sony meghajtó (CRX0811) a különböző sebességgel írt CD-formátumokat nem mindig olvasta zökkenőmentesen, és nem volt elég gyors, azonban a specifikációkban leírt tulajdonságoknak megfelelt. Sajnos a CDSpeed-99-cel nem tudtunk írási sebességet mérni, de ez nem feltétlenül a meghajtó hibája. Kicsit zavaró, hogy elég lassan ismeri fel a lemezeket, ezért viszonylag sokat kell várni, amíg bejön a CD tartalma.

- TDK

A TDK készítette a világ első 16-szoros CLV íróját, amely bejelentésekor a leggyorsabb volt a piacon, megelőzve a konkurens Yamaha és Plextor termékeket. A TDK felhasználta a CyClone sorozatban a Sanyo CRD-BP1400P nevű megoldását. A meghajtó gyorsan és biztonságosan írt, a BURN-Proof technológiának köszönhetően akkor sem rontotta el a lemezt, ha viszonylag jobban terheltük a számítógépet. A tesztben szerepelt készülékek közül mi ezt láttuk a legjobbnak.

- Teac

A Teac az egyik legelismertebb meghajtógyártó, köszönhetően kiváló ár/teljesítmény mutatójú termékeiknek. Magyarországon is ez az egyik legkedveltebb márka. Az egész mezőnyben a Teac írók elérési ideje volt a legjobb, és kiemelkedtek processzorterhelési mutatóikkal is. Mindkét tesztelt készülék jó választás, nemcsak a kedvező ár miatt, hanem mert nagyon biztosan tudják, amit tudnak.

- Yamaha

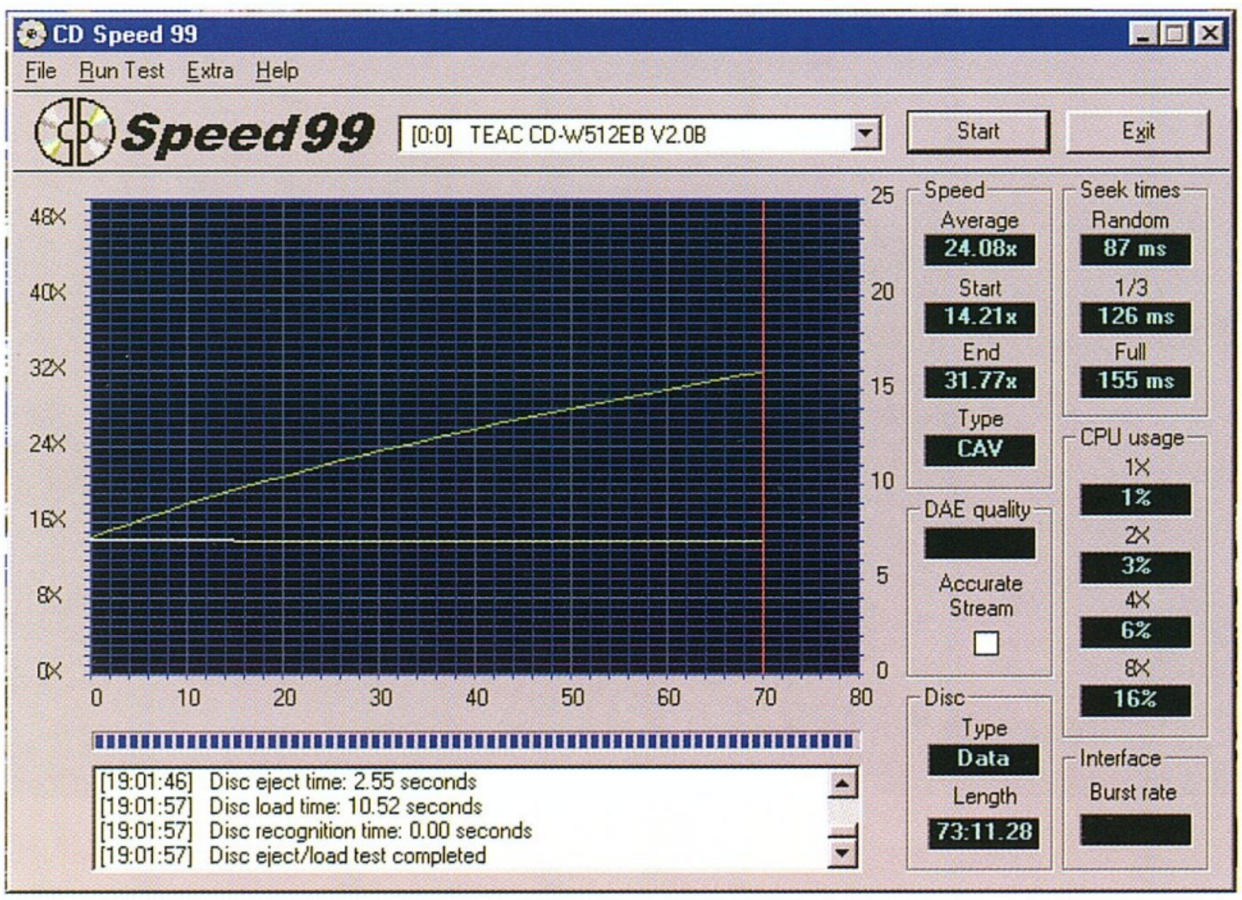
A Yamaha nagy sikerű CRW2100E sorozatának IDE-s képviselője a mezőny egyik legfürgébb tagja. A CRW2100-as sorozat arról híres, hogy ezek voltak a világ első 16x-os írói P-CAV módban. A Yamaha egység hiányossága, hogy nem építettek bele puffertelítetlenség (buffer underrun) elleni védelmet. Ezt a szakemberek azzal indokolták, hogy az óriási, 8 MB-os puffer mindenhez elegendő. A magyarázat azonban nem volt elég meggyőző, amit az is bizonyít, hogy az idei CeBit-en bemutatott CRW2200-as szériát már felszerelték ilyen technológiával. Mindamellett otthoni használatra a 8 MB-os puffer tényleg bőven elegendő.

Az írási metódus alatt több programot is futtattunk, hogy megzavarjuk a működést, de gonoszkodásaink nem jártak eredménnyel. A teszt során az általunk használt márkájú CD-R lemezt nem volt hajlandó 12x-esnél gyorsabban írni (ennek oka, hogy nem szereti a zöld lemezeket, és erre a magyar leírás figyelmeztet is), ezért a dobozban mellékelt Yamaha CD-R lemezzel is teszteltük, és azzal már nem is tapasztaltunk hasonló problémát. Sajnos a mellékelt szoftver nem engedélyezett 8x-osnál gyorsabb újraírást (emiatt került a táblázat megfelelő sorába a 7,1x-es bejegyzés). Más íróprogramokkal is próbálkoztunk, de ezekkel sem jutottunk jobb eredményre. A gyártó honlapján sem találtunk utalást a problémára, ezért megnéztük az íróprogram honlapját, ahonnan kiderült, hogy az újabb Easy CD Creator (5.x) verziók már működtetni tudják a meghajtót.

Kicsit zavaró, hogy a készülék elég hangos, lassan ismeri fel a lemezeket, és lustán pörög fel (sőt néha kétszer is egymás után), azonban ha egyszer beindul, akkor nincs megállás, grabbelésben nem sok meghajtó vette fel vele a versenyt. A Storage System jóvoltából magyar kézikönyv is van hozzá.

A tesztről részletesebb információk is találhatók a (http://plusabit.tvnet.hu) honlapon.

Simon Zoltán
hactor@uze.net



Tesztkonfiguráció

AMD Duron 600 MHz Socket A
Soltek SL75KV Socket A alaplap
Asus V3800 TNT2 M64 32 MB SDRAM
128 MB PC133 SDRAM
Quantum Fireball Plus LM 20 GB (7200 rpm)
LG Flatron 795FT 17" monitor
Windows 98 operációs rendszer

A gyártók honlapjai

www.tdk-europe.com
www.yamaha.co.jp
www.teac.com
www.sony-cp.com
www.freecom.com


Az üzleti kommunikációban mi biztosítjuk a megoldásokat.
Ön nyugodtan nézhet másfajta izgalmak után!

A puffer kényes pont

Tippek és trükkök a CD-íráshoz

CD-íráskor a leggyakrabban előforduló hibajelenség, hogy leáll az írási művelet. A következmény egy elrontott lemez. Ennek oka többnyire az, hogy kiürül a puffer. A buffer underrun, magyarul puffertelítetlenség vagy tárkiürülés tényleg gyakori, de nem mindenre ez a magyarázat. Nézzük meg, miről is van szó.

Olvasáskor a CD-ről egy belső pufferen keresztül jutnak el az adatok a gépben a megfelelő helyre. Íráskor is a puffer közbeiktatásával közlekednek a bitek, csak az áramlás iránya fordított: a buszon keresztül bekerülnek az átmeneti tárolóba (a pufferbe), ahonnan a meghajtó kiveszi őket és rögzíti a CD-re. A buffer underrun (puffertelítetlenség, pufferkiürülés, tárkiürülés) jelensége akkor lép fel, ha a csatornán nem érkezik elegendő adat, ezért a meghajtó kénytelen felhasználni az átmeneti tár teljes tartalmát, és hirtelen nincs mit írnia. Hogy ez miért baj? Mert a meghajtókat a szituáció sokkolja, megakasztja az írási folyamatot. Ahhoz ugyanis, hogy ez ne következzen be, a meghajtónak le kell állítania a lézert, meg kell jegyeznie, hogy hová írta be az utolsó bitet, majd amikor megint jönnek az adatok, akkor az írófejet ugyanarra a helyre kell pozicionálnia, végül bekapcsolni a lézert, és akkor az írás folytatódik.

BURN-Proof

A fenti műveletsorra a meghajtók eleinte nem voltak képesek. A probléma megoldására először a Sanyo mérnökei dolgozták ki a BURN-proof (Buffer UnderRuN Proof) technológiát. A trükk lényege, hogy az íróegység figyeli a puffer tartalmát, és ha az bizonyos telítettségi szint alá csökken, akkor leállítja a műveletet, de elmenti a fej pozícióját. Az írást mindaddig nem is folytatja, amíg meg nem telik a tár. Újraindításkor a pozicionálásnak nagyon pontosnak kell lennie, mert nem szabad, hogy az olvasásnál észrevehető maradjon az adatfolyamban bekövetkezett törés.

Azóta más cégek is kidolgozták a puffertár kiürülése elleni védelmet, annak legfeljebb más az elnevezése: JustLink (Ricoh), BurnSafe (Yamaha) stb. Ma már a legelelterjedtebb íróprogramok mindegyike ismeri ezeket a módszereket, tehát eredményesen alkalmazhatjuk őket.

A tárkiürülés jelensége leggyakrabban akkor fordul elő, amikor túlságosan igénybe vesszük a számítógépet, és nem marad elegendő erőforrás a meghajtó számára. Ha viszont rendelkezünk BURN-Proof írásmódra alkalmas íróegységgel, akkor közben nyugodtan használhatjuk a PC-t más feladatokra is. Ha nincs ilyen technológiánk, akkor CD-íráskor nem érdemes mással is foglalkoztatni a gépet, mert ennek egy-egy elrontott CD lehet az ára.

Nem csak a puffer...

Persze nem csak emiatt szaporodhat elrontott lemezeink száma. Az egyik alattomos hibaforrás az IDE-kezelő. Ha az IDE driver nem tudja megfelelően kiszolgálni a meghajtót, akkor az adatok olvasása és írása rendkívül bizonytalanná válik. A probléma meglétét legegyszerűbben a CDSpeed99-cel ellenőrizhetjük: ha az adatolvasási sebesség és a forgási sebesség ingadozik, akkor van okunk gyanakodni. Főleg Windows 2000-et vagy Millenniumot használók szoktak panaszkodni, hogy a legkiválóbb íróberendezések is elrontják a lemezírást. Erről általában a kiforratlan driverek tehetnek.

Viszonylag ritka eset, hogy az írószoftver okozza a hibát. Ráadásul ennek pontos kiderítése szinte lehetetlen. A szoftvergyártó honlapján esetleg választ találunk kérdéseinkre.

Problémák forrása lehet az is, hogy több lemezt írunk egymás után, és az íróegység túlmelegszik. Egy kicsit pihentetni kell a készüléket.

Amire érdemes figyelni

— Ha IDE-s írót használunk, azt lehetőleg a másodlagos (secondary) IDE buszra csatlakoztassuk, mert így az elsődleges (primary) buszon lévő merevlemez nem akadályozza az egység működését (csökken a tárkiürülés veszélye).

— A DMA használata elengedhetetlen a nagy sebesség eléréséhez, sőt DVD-s egységek esetén a filmlejátszás lehetetlenné válik a nagy processzorterhelés miatt.

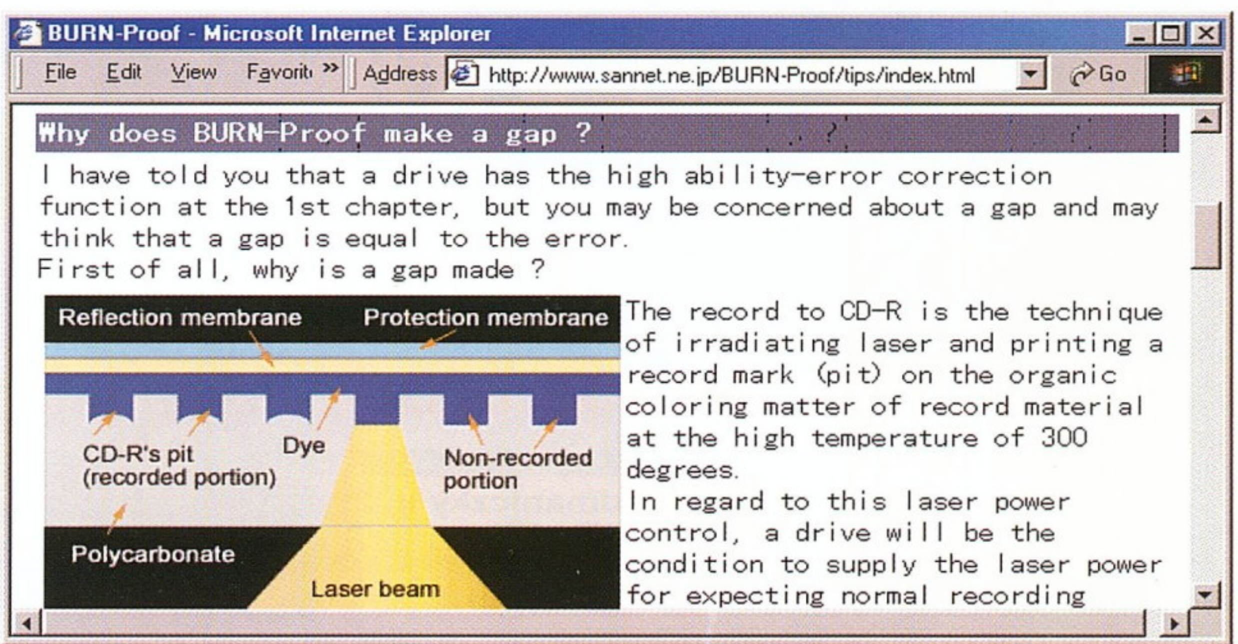
— Az írási művelet megkezdése előtt érdemes megnézni a meghajtó specifikációját, amelyben a gyártók rögzítik, hogy milyen lemezekkel működik együtt az egység. Nem mindegy, hogy 100 forintos „no name", vagy 500 forintos márkás korongot használunk.

— Egyes CD-író egységek nem képesek ugyanolyan sebességgel vésni az azonos típusú, de különböző anyagú lemezeket (gondoljunk csak az írási oldalon arany, ezüst, kék, zöld stb. színű lemezekre).

— Az írási művelet megkezdése előtt, ha az íróprogram engedi, érdemes lefuttatni egy írástesztet.

— Általában lehetőség van úgynevezett image-fájl készítésére, amely a készülő CD teljes tartalmának másolata. Ebből biztonságosabban tud az írószoftver dolgozni, és lényegesen csökken a CD elrontásának valószínűsége.

Simon Zoltán



Digitális kamerák

A videózás új útja



A digitális fényképezés már elég gyorsan terjed, a hobbi szintű modellek a bevásárlóközpontok polcain is megjelentek. Mozgóképet rögzítő társaik még nem tartanak itt, nemcsak az áruk miatt, hanem mert a velük készült felvételek feldolgozásához nagyobb számítógépre és bizonyos szaktudásra is szükség van. Kipróbálásra a Panasonic Magyarország bocsátotta rendelkezésünkre az NV-DS12EG típusú DV kamerát.

A digitális fényképezőgéppel készült felvételek feldolgozására gyakorlatilag bármilyen PC megfelel, a képek mérete tömörítés nélkül is csak néhány MB. A számítógépet pedig akár teljesen kiiktathatják azok, akik megelégszenek a felvételek módosítás nélküli nyomataival, mert már kaphatók olyan nyomtatók, amelyek a fényképezőgép kártyáját beolvasva közvetlenül kinyomtatják a képeket.

A digitális video (DV) kamerákkal más a helyzet. Azokhoz szükségesek a digitális szerkesztőrendszerek, hiszen ha analóg szalagra visszük át az anyagot, elveszítjük azokat az előnyöket, melyek miatt a költségesebb digitális kamerát beszereztük.

A Panasonic NV-DS12EG külső méretében a VHS-C kameránál valamivel kisebb, súlyra is közel azonos, de a kamera DV kazettájának mérete csak töredéke a VHS-C szalagénak, egy ingzsebben is jól elfér. A DV kazetta 4-6-szor többe kerül, mint az analóg kazetta, felbontása viszont több mint duplája, és a hang minőségi paraméterei is sokkal jobbak.

A Panasonic DV kameráinak sorában a NV-DS12EG a legkisebb modell, de a sorozat nagyobb tagjai (15EG, 150EG, 25EG) optikailag azonos képességűek, csak szolgáltatásaikban térnek el az alapgéptől, például van színes LCD nézőkéjük, amelynek felbontása nem túl nagy, érezhető a szemcsézettség az analógoknál megszokott képhez viszonyítva. A soros áttöltéshez külön megvásárolható a WV-DTA9E kábel és a szoftvercsomag (kb. bruttó 14 ezer forintos áron).

A szoftver le tudja menteni a mozgókép-snittek közötti állóképeket vagy a lejátszáskor megállított képeket. Éjszakai felvételekhez beépített infralámpa ad segítséget, ilyenkor a rögzítés fekete-fehérben történik. Mód van állóképek rögzítésére felvétel közben is, egyedi képkockaként, vagy sorozatban a gomb lenyomva tartásáig. Az állóképes kamerákkal szemben itt nincs memóriából háttértárba (flash kártya) írási idő, ezért a sorozat akár az egész kazettát kitöltheti, plusz a képek alatt a hangokat is hallhatjuk. A használható effektek közül a tükrözés, a stroboszkóp és a bűnügyi filmekből ismert „kockásítás" érzékelteti a digitális feldolgozásban rejlő lehetőségeket, ennek inkább csak demonstráció a célja, az anyagot számítógéppel célszerű szerkeszteni, a vágórendszerek sokkal szélesebb palettát nyújtanak.

Egyértelműen a DV felé vezet az út, az ártól eltekintve nincs semmilyen hátránya az analóg technológiához képest. (A hagyományos videón való lejátszáshoz is bármikor készíthetünk másolatokat az analóg kimeneten.)

A Panasonic fenti kamerasorozata mellett vannak olyan modellek is (például a Panasonoc NV-MX300), amely a memóriakártyára nagyobb felbontásnál rögzíti az állóképeket, mint a videofelvétel, így felesleges fényképezőgépet is cipelni. Másrészt viszont valamivel olcsóbban kaphatók hibrid technológiás Digital8 kamerák is, amelyek képesek egyrészt Hi8-as szalagra digitálisan felvenni, másrészt az analóg Hi8-as filmeket a digitális kimeneten lejátszani, így nem kell a digitalizálást a számítógépen külön elvégezni.

Bánó György

Házimozi a számítógépben

Talán nem mindenki tudja, hogy a számítógépbe beépített DVD-ROM segítségével éppúgy nézhetünk DVD-n rögzített filmeket, mint egy asztali DVD-lejátszóval, különösen ha a számítógép monitora helyett nagyméretű ernyőn nézzük a projektorral vetített képet.



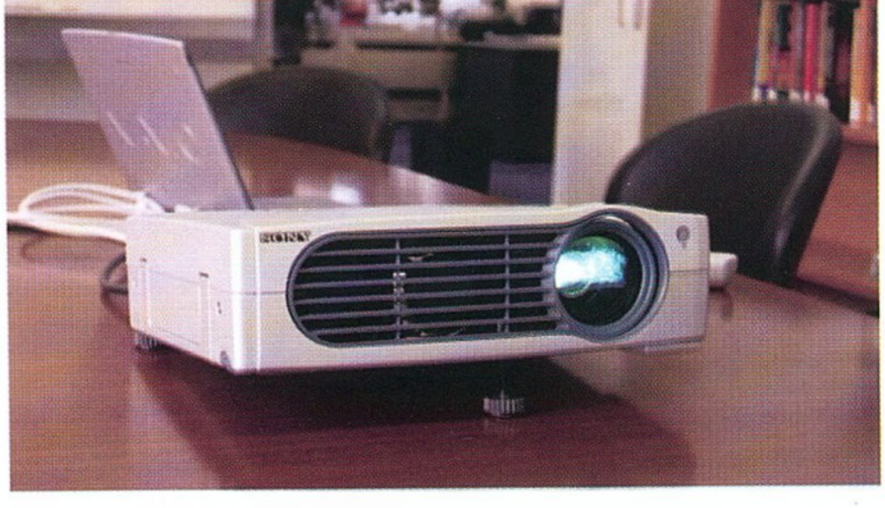
A DVD-filmek lejátszásának egyik alternatívája kezdettől fogva a számítógép. A hazai piackutatások szerint Magyarországon is elég sokan használják erre a gép DVD-ROM egységét. Azzal érvelnek mellette, hogy egy jobb minőségű DVD-lejátszó árából megoldható a DVD-lejátszás, és emellett van egy számítógépük is. Akik inkább az asztali DVD-lejátszót választják, kényelmetlennek tartják, hogy a számítógépet össze kell állítani, az nem mindig működik üzembiztosan, kicsi a képernyője, sok szolgáltatás hiányzik belőle stb.

Vajon melyik produkál jobb minőséget DVD-lemezről, a számítógép vagy az asztali lejátszó. Erre a célra egy teljesítményben és árban is kiváló vetítőt, a Sony VPL-CS10 projektort választottunk. A jelenlegi projektorok közül a VPL-CS10 mind árban, mind minőségben komoly alternatívája például egy 60 colos (150 centiméteres) hátulról vetítő projektornak (retro doboz). Ez utóbbi ára összemérhető a Sony VPL-CS10 árával, sőt néhány típusa jóval drágább.

A projektor képmérete könnyűszerrel növelhető akár a 2 méteres képátlóig, vagy azon túl is. A fényerővel és a kontraszttal lehetnek gondjaink, de a felbontással semmiképpen sem. Az LCD projektorok, így a Sony VPL-CS10 előnye, hogy nem kell törődnünk az 50 vagy 100 hertzes megjelenítéssel, hiszen az LCD panel vetítési technológiájából adódóan a kép szinte egyáltalán nem villog, még nagy képátló esetén sem.

Ráadásul a Sony VPL-CS10 számítógép-bemenetét használva (megfelelő számítógépes videokártyával és szoftverrel) a számítógépek monitorkimenetén sokkal jobb képminőség érhető el, mint az egyszerűbb kivitelű, RGB kimenet nélküli asztali DVD-lejátszókkal. Ennek oka elsősorban az, hogy a jelenlegi televíziós szabványok képfelbontása a legjobb esetben (PAL szabvány esetén 720x576 képpont) sem éri el a számítógépeknél lassan már „gyengécskének" számító SVGA (800x600 képpontos) felbontást. (A képminőségbeli különbség igen jól látszik, ha egy S-Video kimenettel rendelkező számítógépet egy televízió S-Video bemenetére csatlakoztatunk: egyébként jól látszódó szövegek is olvashatatlanná válnak.)

A másik lényeges számítógépes felhasználási terület a játék. Az egyik számítástechnikai lap nemrégiben tesztelte a PlayStation 2 teljesítményét, és azt tapasztalták, hogy a mostani játékok még közel sem aknázzák ki mindazt, ami a „vasban" rejlik. A Sony VPL-CS10 lehetővé teszi, hogy a számítógépes játékokat ne csak egy 15–17 colos monitoron élvezhessük, hanem „életnagyságban" és a hagyományos televízióénál nagyobb felbontásban.



A Sony VPL-CS10 kétméteres képátlója más 3D grafikához, PowerPoint prezentációhoz, vagy más üzleti bemutatóra is jól használható. Ami a Dolby Digital (vagy DTS) hangot illeti, a digitális kimenettel rendelkező hangkártyák között már vannak Dolby Digital kompatibilisak, és vásárolhatunk DTS-re felkészített külső dekódolót is, akár 40 ezer forint körüli összegért.

A DVD filmek lejátszására próbaképpen használt számítógép konfigurációja a következő volt: PIII 850 MHz processzor, 256 MB RAM, ATI Radeon All-in-Wonder 32 MB DDR RAM, Pioneer 10x DVD-ROM, SoundBlaster Live hangkártya. Mind az ATI videokártya, mind a SoundBlaster rendelkezett Dolby Digital kimenettel. A számítógép monitorkimenete össze volt kötve a projektorral, 800x600-as felbontást használtunk, 32 bites színmélységgel. A beállítások után indult az ATI DVD lejátszó programja.

Az első film, amelyet számítógépes DVD-lejátszással megnéztünk, az „Egy bogár élete" volt. A képminőség különbsége azonnal megmutatkozott. A számítógép élesebb, részletdúsabb, dinamikusabb képet produkált, különösen azoknál a jeleneteknél, ahol sok hangya (bogár) látszik egyszerre. Rengeteg apró részlet van a felvételen, amelyre korábban fel sem figyeltünk. Még a hátsó sorokban álló bogarak alakja is jól felismerhető. Kipróbáltuk a képkülönbséget a „Briliáns csapdával" is, hogy ne csak animált képeket láthassunk. A hatás hasonló volt: élettelibb, elevenebb képek.

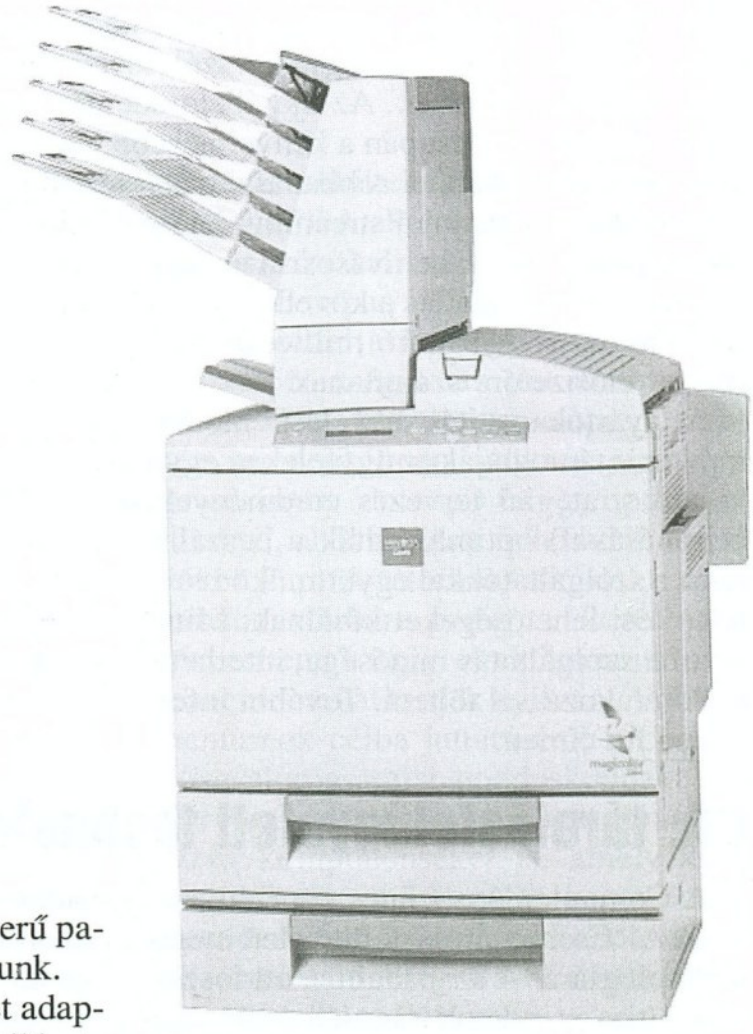
A játéklehetőségeket sorra véve a Need for Speed Porshe Unlished játékot vettük elő, amelyet eddig nem játszottuk 19 colosnál (50 centiméteresnél) nagyobb képátlón. A Sony VPL-CS10 itt is felülmúlta várakozásainkat.

A modern projektortechnika komoly kihívást jelent a hátulról vetítő projektorok és képcsöves televíziók számára. Kihasználva a monitorbemenet által nyújtott lehetőségeket, a számítógépes DVD-lejátszás (ideális esetben) jobb minőséget produkál egy olcsóbb, középkategóriájú asztali DVD-nél is. Tény, hogy az általunk használt számítógép-konfiguráció ára lényegesen jobb, mint egy asztali DVD-lejátszóé, de ha mellérakunk egy PlayStation 2-t, és azt nézzük, hogy milyen előnyöket nyújt egy vetítő egy projektoros televízióhoz képest, akkor mindenképpen a VPL-CS10-zel járunk jobban. Az ilyen jellegű összeállítás persze azoknak ajánlható, akik nem riadnak vissza egy program vagy egy videokártya telepítésétől, de az ő számukra mindenképpen megfontolandó lehetőség.

További információk és tesztvetítés: www.sony.hu/presentation. **Érdeklődés telefonon: 270-8526.**

Színt vinni az irodákba

Új Minolta-QMS hálózati lézerprinter



Munkahelye válogatja, hogy milyen mértékben, de a színes nyomtatás lehetőségét az irodák is igénylik. A színes tintasugaras nyomtatók azonban a vállalatoknál nem tudtak a privát szférához hasonló sikereket elkönyvelni, részben kis kapacitásuk, részben a működtetés nagyobb fajlagos költsége miatt, hiszen az irodák „zabálják" a papírt és a tintát. A színes lézernyomtatók előrenyomulását pedig fékezi a magas beruházási összeg, hiába kerülnek később kevesebbe a színes festékkazetták. Az ellentmondást a gyártók új fejlesztésekkel igyekeznek feloldani. A Minolta-QMS speciálisan hálózati használatra kifejlesztett, Crown II technológiájú nyomtatója a nagy teljesítményt ötvözi a gazdaságossággal.

Lehetőségünk nyílt szerkesztőségünkben kipróbálni a Minolta-QMS Magicolor 2200N színes hálózati nyomtatót. Annak konfigurálása kissé időigényes volt, mert a címek és nevek beállítása a menüvezérlő nyilakkal történik, de ezt általában úgyis csak egyszer kell elvégezni. Támpontul szolgált hozzá egy magyar nyelvű füzet és egy ötnyelvű kézikönyv. (A dokumentáció CD-n is rendelkezésre állt.)

A kliensek installálását a többi hálózati nyomtatóéval analóg módon lehetett elvégezni.

A hálózati nyomtatók használatának tipikus hátránya (ami felfogható persze előnyének is), hogy irodán belüli népvándorlással jár, különösen a jelentéskészítési időszakokban. A tapasztalat azt mutatja, hogy sok helyi lézernyomtató helyett néhány hálózatosat alkalmazva ez a kényelmetlenség nem borítja fel a munkarendet, viszont jelentősen csökken a karbantartási és felügyeleti költség, továbbá a szem előtt lévő nyomtatón kevesebb „felesleges" oldal jön le, mert mindenki jobban meggondolja, hogy mit akar papíron is látni.

A Magicolor 2200-as sorozat nyomtatási felbontása a legkisebb modellnél is 600x600 dpi, és a nyomat minősége kifejezetten látványos (feltéve, hogy maga az eredeti digitális kép is az). Mivel a színek külön menetben „vasalódnak rá" a papírra, a sok színt tartalmazó részeken sincs egybefolyás. Nemcsak speciális papíron, hanem a lézertechnológiának megfelelő egyszerű papíron is kitűnő nyomatokat kapunk.

A beépített 10/100-as Ethernet adapterrel a nyomtató képes vegyes hálózatban is működni (TCP/IP, NetWare IPX/NDS, kiegészítővel tokenRing), és SCSI interfésszel bővíthető. Szükség esetén rádugható a szokásos párhuzamos portra is, bár akkor nem lehet kihasználni a nyomtató percenkénti 20 oldalas fekete-fehér teljesítményét, és megtölteni az alapmodellnél is minimum 64 MB-os memóriát. A színes nyomtatás „csak" 5 lap percenként, a készülék rotálja a tonereket, így egyszerre egy szín felvitelére kerül sor.

A megadott sebesség viszont valóban reális, még fotókat nyomtatva is, szemben azzal, hogy a tintasugaras nyomtatók értékei általában a draft (vázlatos, alacsony felbontású) módra vonatkoznak, és sokszor ott is csak az egyszerű szövegek kinyomtatásakor közelítik meg azt.

A beépített mikroprocesszoros feldolgozás és a memória mérete lehetővé teszi, hogy amíg az egyik feladat nyomtatása folyamatban van, addig fogadni tudja a következőt, így minimálisra csökken a várakozás. A használható leírónyelvek skálája igen széles. Az opcionális beszerelt merevlemezzel a színkalibrációs módokat is használhatjuk. A papírtároló kapacitása 1150 lapig bővíthető, ami erőteljes igénybevételnél is napokig elég. Ugyancsak opcionálisan kapható a kétoldalas nyomtatáshoz szükséges modul, valamint felszerelhető a nyomtatóra a felhasználók anyagait szétválogató, ötrekeszes szortírozó.

A nyomtató szoftvere a monitorozást leegyszerűsíti, követhető vele a kellékek, a tonerek állapota, figyelmeztet a kötelező karbantartásokra, a felhasználók ezenkívül a web alapú CrownView segítségével is ellenőrizhetik az állapotát. A Print Auditorral a hálózati nyomtatók költségnyilvántartása táblázatba foglalva kezelhető.

A 2200N forgalmazásának bevezető akciója keretében az alapmemóriát megduplázták (128 MB), és a fekete tonert egy évig ingyenesen adják. Számításaim szerint ez utóbbi akár százezer forint feletti összeg is lehet, ha valóban megfelelően kihasználják a nyomtatót. A nettó végfelhasználói ár 694 990 forint. Aki irodai szinten számol, az tudja, hogy egy nagy teljesítményű fekete-fehér nyomtató ára hálózati kártyával 300 ezer forint körül van, és a festékkazetták is 25-30 ezres tételek, ezzel összevetve pedig már a színes hálózati nyomtató ára sem tűnik olyan nagy összegnek. Természetesen a befektetés csak ott térül meg, ahol havonta több ezer oldalt nyomtatnak, megfelelő a hálózati infrastruktúra — és van mit színesben kinyomtatni.

Bánó György
gyorgy.bano@alcoa.com

3Com: sikeres kábelmodemek

A 3Com bejelentette, hogy világszerte több mint 1,2 millió DOCSIS-kompatibilis, kétirányú külső kábelmodemet értékesített a kábeles és nagy sávszélességű internetelérést szolgáltató ügyfeleinek. Az új kábelmodemes internet-előfizetők számát nem csupán a kényelmesebb internetezés iránti kereslet növeli, hanem robbanásszerűen terjedő újdonságok egész sora is, például a streaming audio és video, a digitális fényképezés stb. A behívásosról a nagy sávszélességű internetelérésre való átállás a következő években tovább gyorsul, 2004-re az USA-ban 16 millió, és Magyarországon is több tízezer előfizetőre számítanak.

A gyártókapacitás növelése mellett itthon két további módon is támogatják az ügyfeleket: egyrészt az egész iparágra kiható stratégiai tervezés eredményeként (és a beszállítók bevonásával) optimalizálták a beszállítói csatornákat, másrészt a szolgáltatókkal együttműködve egyszerű, automatikus telepítési lehetőségeket kínálnak. Mindezek eredményeként javul a szolgáltatás minősége, internetezéskor kevesebb időt kell várakozással tölteni. További információk a http://www.3com.hu címen.

CA: tárolásfelügyeleti technológia

A Computer Associates nemzetközi szabadalmat szerzett a CA ARCserve tárolásfelügyeleti megoldásában alkalmazott technológiára. A szabadalmazott technológia automatikusan biztosítja az adatok tartaléktárolását a helyi hálózatokon, valamint lehetővé teszi az adatok felhasználását heterogén rendszerekben. Az elektronikus üzletvitel számára tervezett ARCserve szoftver alapú tárolásfelügyeleti megoldás, amely szabadon skálázható a helyi (SAN, LAN), illetve nagy kiterjedésű (WAN) hálózatokon használt megoldásokhoz. Az ARCserve-ről és a hozzá kapcsolódó applikációkról további információk szerezhetők a www.ca.com/arcserve címen.

Intel: hálózati processzorok

Az Intel új, gyorsabb hálózati processzorral és robusztusabb szoftvereszközökkel erősítette meg az Internet Exchange Architecture (IXA) nevű hálózati architektúráját. Egy éven belül ez a második jelentős fejlesztés mind az Intel IXP1200 hálózati processzorcsalád, mind a hozzá kapcsolódó szoftverfejlesztő eszközök területén. Az Intel új szoftverfejlesztő készlettel (SDK) és az új Microengine C fordítóval segíti elő a sokféle szolgáltatást nyújtó hálózati eszközök fejlesztését. Az IXP1200 hálózati processzor harmadik verziója gyorsabb belső végrehajtó egységekkel és sínekkel rendelkezik. A vezérléstároló egységet a processzor mind a hat mikromotorjában megduplázták, lehetővé téve a fejlesztőknek a még bonyolultabb szolgáltatások beépítését.

Avaya: routing kapcsoló

Az Avaya a hang-, adat- és képátvitelt integráló vállalati hálózatokba szánt adatkapcsolók csúcsmodelljét hozta forgalomba. Az új Cajun P882 MultiService Backbone Switch (több szolgáltatást támogató gerinchálózati kapcsoló) alapkiépítésben (ún. dayone-ready) alkalmas egyidejű adat-, hang- és képátviteli alkalmazásokhoz, és képes akár 128 darab gigabit ethernet, illetve 768 darab 10/100-as felhasználó port kapcsolására. Ezáltal a vállalkozások növelni tudják nagy sebességű konvergens gerinchálózatuk kapacitását, és az egy-egy kapcsolóra rákötött felhasználók számát. A Cajun P882 típusú kapcsoló 139 Gbps kapcsolóteljesítménnyel, és 106 Mpps routolási áteresztőképességgel rendelkezik. A Cajun P882 kapcsolóban felhasználhatók az Avaya korábbi és új 80-as sorozatú hálózati moduljai egyaránt — gigabit Ethernetet, 10/100 Mbps-ot kínálva réz-, illetve optikai hálózatokon, akár 48 portot modulonként. Ez a felhasználók számára lehetővé teszi a magas portsűrűség kialakítását, nagyobb teljesítmény elérését, és a fejlett QoS funkcionalitás kihasználását.

A P882 kihasználja az Avaya web alapú és helyi hálózatmenedzsment alkalmazásai által felkínált lehetőségek teljes skáláját. Támogatja a CajunRules nevű policy management alkalmazást. A Cajun P882 amerikai listaára kb. 25 ezer dollár standard konfiguráció esetén, illetve 32 ezer dollár hibatűrő konfiguráció esetén.

RAD/Lanex: hálózati megoldások

Ilan Seidner, a RAD kommunikációs igazgatója hazai partnerük, a hálózatintegrátor Lanex Kft budapesti sajtótájékoztatóján körvonalazta cégének az elérési (access) hálózatok terén követett termék- és megoldási stratégiáját, három területre koncentrálva: az IP-világára, a mobil telefóniára és a vezetékes szolgáltatásokra. Az IP-világban legalább két megközelítés lehetséges. Az egyik a VoIP forradalmian új technológiája, a másik a TDMoverIP, amely hatalmas megtakarításokat tesz lehetővé, megengedi az új IP alapú transzport hálózatok használatát különösebb kockázat és nagy beruházás nélkül. Az egyre nagyobb sávszélességet és több szolgáltatást igénylő mobil felhasználók rövidesen a második, majd a „két és feledik", illetve a harmadik generációs maroktelefonokkal is találkoznak, miközben a mobilos szolgáltatók szeretnék meglévő infrastruktúrájukat a lehető legnagyobb mértékben megőrizni.

A RAD Data a jelenlegi infrastruktúra mellett biztosít nagyobb sávszélességet. A módszer lényege, hogy TDM helyett csomag (packet) bázisra (ez az ún. 2,5. generációs megoldás), a harmadik generáció esetében pedig ATM vagy IP alapra helyezik a kommunikációt. A vezetékes szolgáltatásban a RAD arra kínál lehetőséget, hogy a távközlési cégek a monopóliumok megszűnésének körülményei közepette is megtarthassák eddigi partnereiket, mégpedig egyazon vezetéken többféle szolgáltatást biztosítva számukra. A RAD megoldása lehetővé teszi, hogy — tipikusan kis és közepes vállalatok számára — ugyanazon az érpáron ún. szolgáltatáscsomagokat lehessen nyújtani. A Lanex ebben kecsegtető üzleti lehetőségeket lát.

IP alapú távbeszélő-alközpont

Már Magyarországon is kapható a 3Com SuperStack 3 NBX üzleti-vállalati IP alapú telefonalközpont, amely az NBX európai követelményeknek megfelelő új változata. Segítségével a cégek akár 600 felhasználóig bővíthetik telefonhálózatukat. A szabványos Ethernet hálózatokon működő SupetrStack 3 NBX főbb jellemzői: egyidejű adat- és hangátvitel; internettelefonálás; böngésző alapú adminisztrációs eszközök; Network Supervisor és NBX NetSet felügyeleti eszközök; VxWorks valós idejű operációs rendszer; 72 beépített hangposta-port; kilenc beépített lokalizált verzió. Az ár a konfigurációtól függ, a 200 db 10/100-as telefont tartalmazó rendszer 140 ezer dollár.

Kovács Attila
akovacs@infopen.hu

Csoportmunka az „egynetben”

A golyóálló GroupWise 6

A vállalati számítógépes rendszerekben szorgalmasan termelt dokumentumok ugyanúgy „maguk alá temethetik” a feldolgozásukkal foglalkozókat, mint a hagyományos irathalmazok. A digitális áradat áttekinthetetlenségének elkerülésére megfelelő módszereket kell alkalmazni, amihez az igazán professzionális rendszerek a dokumentumok puszta nyilvántartása mellett számos egyéb funkciót is felvonultatnak. Ilyen kibővített rendeltetésű szoftver a Novell GroupWise, amelynek korábbi verzióiról már többször írtunk, és amely fokozatos fejlesztéssel címtár alapú rendszerré vált.

A GroupWise kezdeti, elsősorban dokumentumok rendezésére és intranetes dokumentumkezelésre (e-mail, fax, belső levelezés stb.) kidolgozott rendszeréből komplex csoportmunka szoftver lett, amely természetesen továbbra is rendelkezik az alapvető dokumentumkezelési funkciókkal, de mellette sok más képességgel is.

A GroupWise fejlesztésének egyik fő mozgatórugója kétségtelenül a Novell operációs rendszerének folyamatos átalakulása volt (legutóbb natív IP-s NetWare 5-össé). A másik fontos tényező az oprendszeres tartozékból önálló termékké vált címtár, az NDS (Novell Directory Service), amely jelenleg már több operációs platform hálózati üzemeltetőinek kedvelt natív eszköze. Ahogy ez utóbbi fejlődött, úgy alakul át napjainkban is a Novell termékkínálata, és abban egyre nagyobb szerepet töltenek be az NDS-re, újabban pedig az eDirectoryra alapozott alkalmazások. A folyamatot jól jelzi, hogy a „gyári” fejlesztőműhelyben új fejlesztési koncepciót fogalmaztak meg: már nem a Novell korábbi sikertermékét, a hálózatos operációs rendszert, hanem a címtárat helyezik a középpontba. Ezt a koncepciót az internetesedő világnak megfelelően a „One Net” elnevezéssel fejezték ki.

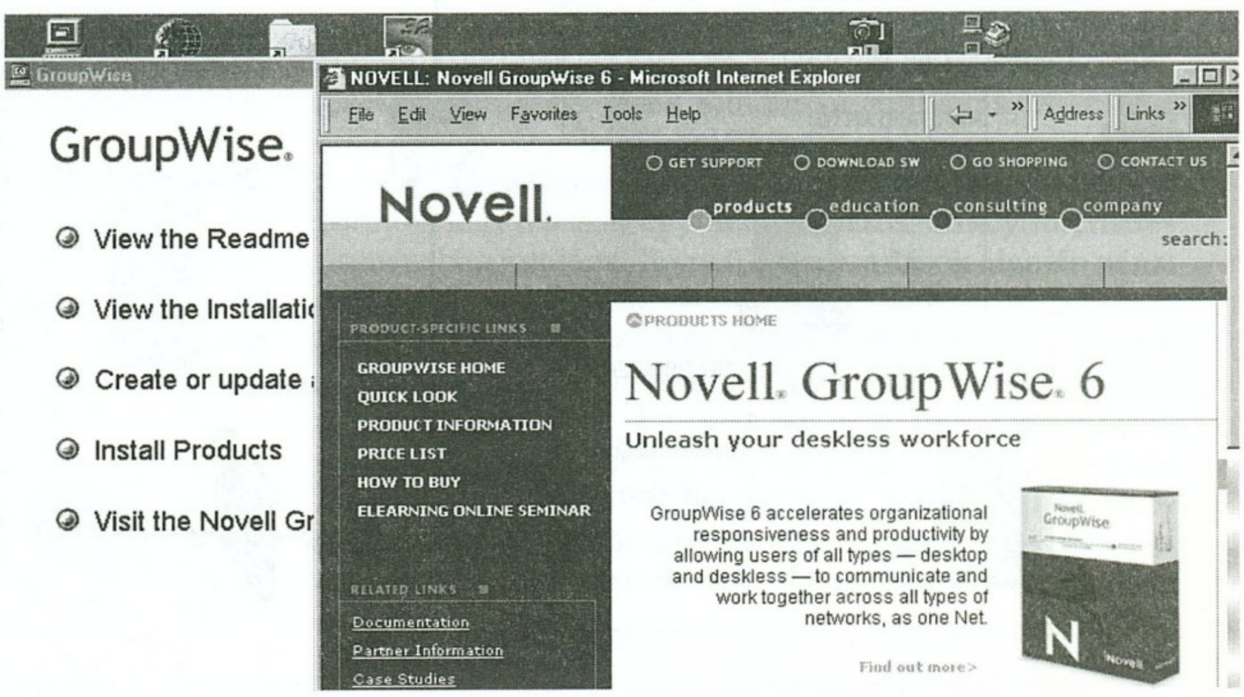
A GroupWise verzióváltásának előkészítése ugyanennek a koncepciónak a keretében folyt. A BulletProof (golyóálló) néven kifejlesztett 6-os verzió idén tavasszal készült el, és a CeBIT-en mutatkozott be a nagyobb nemzetközi nyilvánosság előtt is. A címtár alapú adatkezelésnek köszönhetően az új alkalmazás valóban „golyóállónak” bizonyulhat a gyakorlatban, hiszen az NDS kapcsán nem nagyon hallunk eredményesen végződött feltörésekről. A hozzáférési jogosultságok kezelését illetően a GroupWise feltehetően szintén örökli a megbízhatóságot az elektronikus levelezés, a naptári szolgáltatások, a dokumentumkezelés vagy a munkafolyamat szervezésének megvalósításakor.

Van továbbá a Novell új csoportmunka-alkalmazásának olyan jellegzetessége is, amihez vetélytársainak többnyire „külső segítségre” van szükségük. Ez pedig a napjainkban tért hódító mobil informatika beépített támogatása. A központilag menedzselhető egységes rendszer bárhonnan és bármilyen mobil eszközről elérhetővé tehető az arra jogosultak számára. Ez azt jelenti, hogy a GroupWise szolgáltatásai az adott mobil eszköz technikai korlátait figyelembe véve használhatók távolból is, például mobiltelefonnal vagy tenyérgéppel (PDA, Personal Digital Assistant eszközökel), amilyen a PalmPilot vagy a Compaq iPack.

Tekintettel arra, hogy a GroupWise alapvetően mégiscsak dokumentumok kezeléséhez készült, az alaprendszerhez mindenképpen szükség van egy tartományra (domain) és egy „postahivatalra”, ahol minden felhasználó saját fiókot kap. A létrehozandó dokumentumkönyvtárban pedig a kezelendő, kézbesítendő vagy elraktározandó dokumentumokat tartják nyilván. A dokumentumok célba juttatásáról a GroupWise alprogramjai gondoskodnak. Az alprogramok között találunk olyan további komponenseket is, amelyekkel megoldható az internetes kapcsolatok menedzselése, vagy a rendszer további részeivel való integrálás. Ez utóbbi révén igénybe tudjuk venni a cég hardvereszközei közül azokat is, amelyek nem NetWare oprendszert használnak, de értik az NDS-t. Ezáltal a dokumentumkezelés és az ezen alapuló összes tevékenység vállalati szinten is könnyebben összehangolható.

Az is tény persze, hogy ez utóbbi integrált megoldásra szükség is van, hiszen a Novell jelenleg (sajnos) nem fejleszt kliensgépeken futtatandó saját operációs rendszert, azokra emiatt többnyire 32 bites Windowst telepítenek, míg az adminisztráció gyakorlatilag platformfüggetlen módon megvalósítható a Java alapú, a Java virtuális gépén (JVM) futó ConsoleOne segítségével.

Simay Endre István



Az online tranzakció biztonsága

Valódi üzletkötés virtuális környezetben

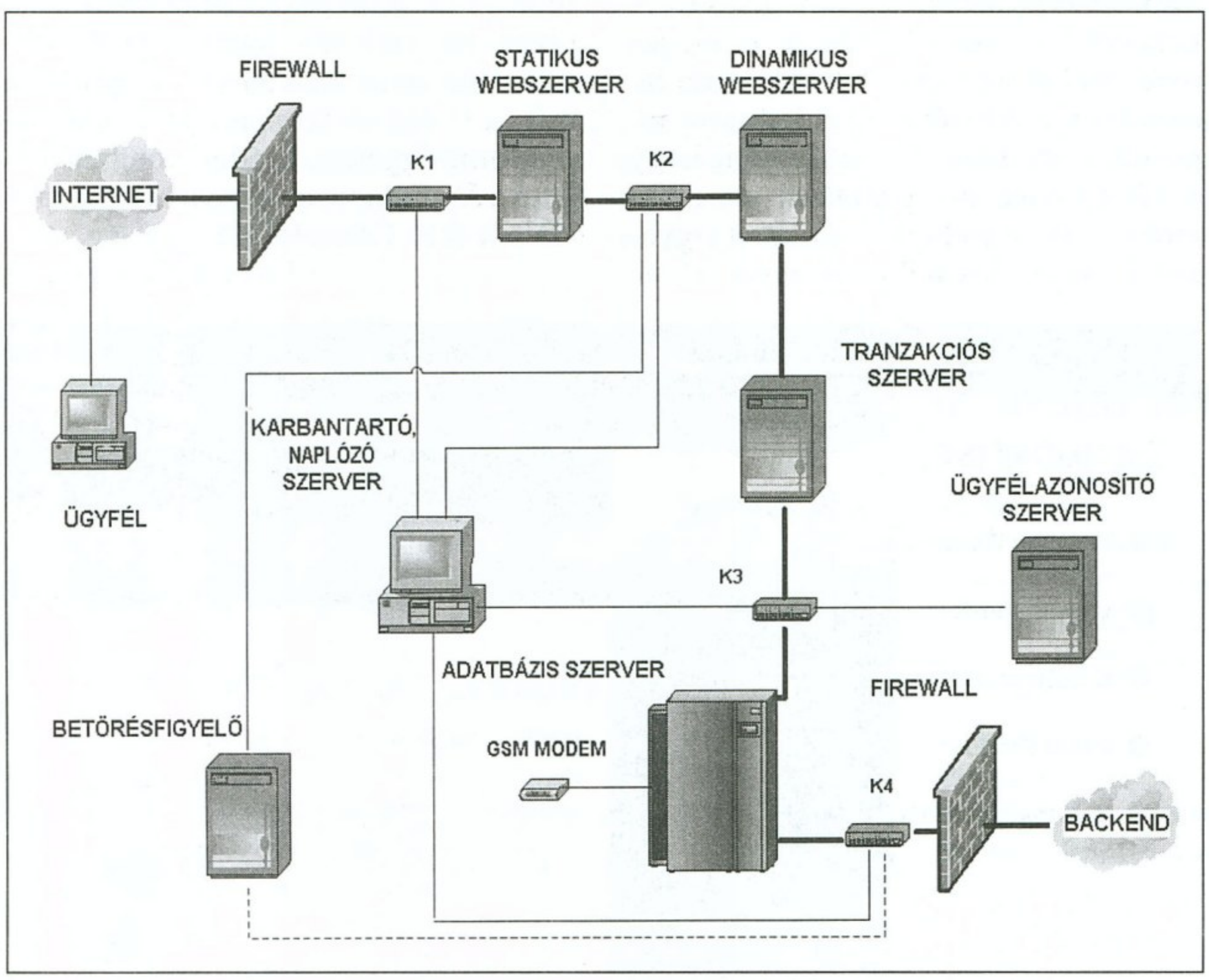
Sikeres és kudarcot valló próbálkozások hosszú során kell átesni, mire kialakulhat, hogy a vásárlásnak mely elemei milyen mértékben terelődnek majd át stabilan az elektronikus csatornákra. Az viszont egyre nyilvánvalóbb, hogy az egyik kulcskérdés az ügyintézés mozzanatainak hiteles dokumentálhatósága, a másik pedig az elektronikus fizetés megbízható rendszere lesz. Ezért fontos lépés a digitális aláírás bevezetése, és ezért különleges jelentőségű az internet alapú banktechnika.

A hagyományos vásárlás kissé elfedi azt, hogy a vevőn és az eladón kívül mindig részt vesz a tranzakcióban egy harmadik szereplő is: a pénz képében jelen lévő bank. Az internetes vásárlásnál a fizetéshez azonban már a vevőnek is, az eladónak is szüksége van a bank aktív közreműködésére. A vevőnek ugyanis hiteles bizonyítékkal kell rendelkeznie arról, hogy valóban a pénzintézettel kommunikál, amelyre a tranzakciót rábízhatja. Az eladónak pedig tudnia kell, hogy nem „virtuális vevő" van a túloldalon, hanem az valóban fizet, és ezt a bankrendszer is garantálja.

Hagyományos ügyfélkapcsolat esetén az ügyfél személyesen megjelenik a bankban, ahol megvannak azonosítási adatai, aláírásmintái. Az ügyfél számára ugyanitt kellő garancia, hogy egy fizikailag létező bankfiókban van. Teljesen online kapcsolat esetén a banknak egészen más eszközöket kell alkalmaznia az ügyfél személyazonosságának ellenőrzésére, ami egyébként a bank törvényi kötelezettsége is.

Ennek a megoldására jöttek létre a hitelesítő hatóságok (CA, certification authority), amelyek saját aláírásukkal teszik hitelessé az ügyfél aláírási kulcsát.

Lehetővé vált tehát a virtuális bankok megjelenése az interneten. A bankok iránti bizalmat ebben a kapcsolatrendszerben csak a biztonság megteremtésével és megismertetésével lehet kialakítani.



A biztonság bástyái

Az internetes banki rendszerek biztonsági szolgáltatásai általában az alábbi elemekből épülnek fel:

— Rendszerbiztonság (security)

A rendszerbiztonság az illegális hozzáférési kísérletek elleni védelemre szolgáló eszközöket és eljárásokat foglalja magában. Kulcselemei a megfelelően konfigurált hálózati tűzfalak, valamint a behatolásdetektáló (intrusion detection) eszközök.

— Azonosítás (authentication)

Az azonosítás a kommunikáló felek kölcsönös személyazonossági vizsgálatának feltételeit teremti meg. A bankok általában szimmetrikus titkosítási technológiát használnak az üzenetek titkosítására, és aszimmetrikus (nyilvános+titkos kulcsú) kriptográfia szolgálja az azonosítást. Aszimmetrikus kriptográfia céljára a maximum 1024 bites RSA eljárás terjedt el széles körben. Bár a biometrikus azonosítás (retina, ujjlenyomat, arcforma, hang alapján) nagy előrelépés lenne, széles körű elterjedésére elsősorban a szükséges infrastruktúra költséges volta miatt még várnunk kell.

— Megbízhatóság (trust)

A megbízhatóságot a nyilvános kulcsú titkosításra épülő rendszerekben egy harmadik fél, a hitelesítő szervezet garantálja. A kölcsönösen elfogadott hitelesítő szervezet mindkét oldal, a kliens és a banki webhely számára egyaránt igazolja a másik fél azonosságát.

— Letagadhatatlanság (nonrepudiation)

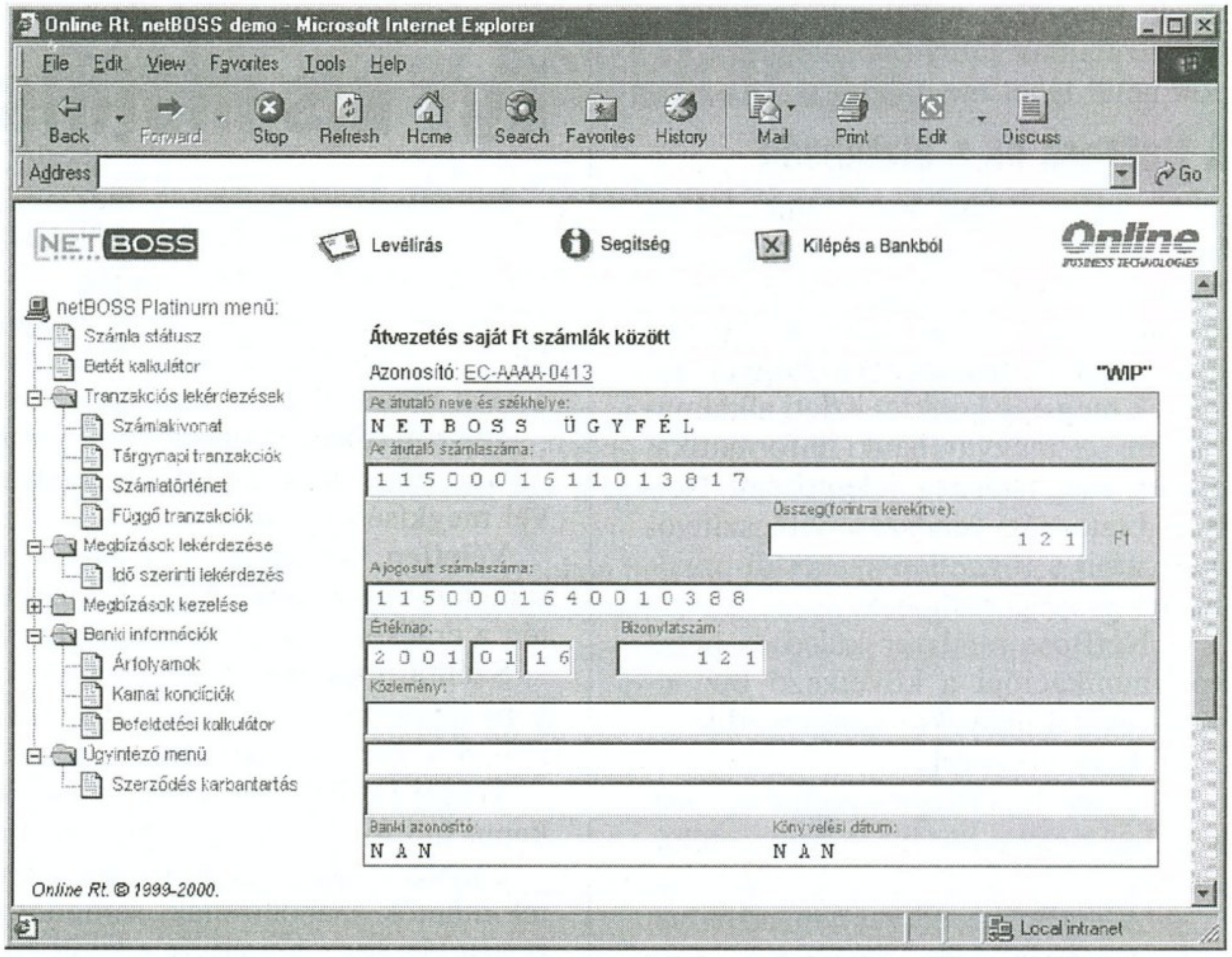
A tranzakciók letagadhatatlansága mindkét félnek érdeke. A PKI technológia erre a célra lett megalkotva, de a jogi szabályozás ebben a kérdésben ma még az Egyesült Államokban sem egységes.

— Adatvédelem (privacy)

Az adatvédelem elsősorban a személyhez kötődő információk védelmét jelenti. A személyes adatok sérelemétől való félelem a kimutatások szerint az egyik kerékkötője az elektronikus banki szolgáltatások terjedésének.

— Rendelkezésre állás (availability)

A 24 órás folyamatos működés biztosításához a vártnál komolyabb erőfeszítések szükségesek. Ezt jól példázza, hogy az USA illetékes felügyeleti szervéhez (SEC, Security and Exchange Commission) beérkező panaszok túlnyomó része az online banki szolgáltatások gyakori szünetelésére, hozzáférhetetlenségére vonatkozik, de Magyarországon is markánsan kiütköztek a folyamatos rendelkezésre állás biztosításának nehézségei.



Intézkedések hálója

Alapszabály, hogy egy rendszer biztonságának megítélésénél a leggyengébb láncszemet kell alapul venni. Ezért nem elég egyetlen alkalmazást, jelen esetben egy internet banking rendszert a környezetéből kiragadva vizsgálni, mert a teljes banki informatikai infrastruktúra biztonságáról kell gondoskodni. A pénzintézeteknél a biztonsági intézkedések rendszerének (security policy) elsősorban az alábbiakra kell kiterjednie:

— A gépek, adathordozók fizikai védelme, a hozzáférésre jogosult személyek körének kijelölése.

— A rendszert használó személyek azonosságának szigorú megállapítása, jogkörének pontos meghatározása.

— Az adatforgalom és az adatállományok védelme az illetéktelenektől kriptográfiai és más módszerekkel.

— Az adatállományok archiválása, mentése.

— Valamennyi rendszeresemény folyamatos (real-time) figyelése, a felhasználók tevékenységének naplózása, a naplók rendszeres elemzése.

— A biztonsági rendszer és a teljes rendszer legalább félévenként történő felülvizsgálata.

A védelem tervezésénél nagy figyelmet kell fordítani arra, hogy nemcsak a külső betolakodók jelenthetnek veszélyt, hanem a bármilyen hozzáféréssel rendelkező belső munkatársak is, ezért a kár megelőzése és a veszély elhárítása a cégen belül fontos szervezési és menedzselési feladat. (A felmé-

Néhány nevezetes (e)bankrablás

1994. június 1. A hongkongi Philippines National Banknál észrevették, hogy eltűnt félmillió dollár a Citibanknál vezetett számlájukról. A pénzt egy finnországi számlára engedély nélkül átutalták, ahonnan az nyomtalanul eltűnt. A Citibanknál először belső visszaélésre gyanakodtak, később kiderült, hogy külső támadás történt. Ez volt az első nagyszabású, eredményes crackertámadás egy bank ellen.

Az elkövetkező hónapokban még 18 alkalommal támadták meg hasonló módon a Citibank készpénzes (cash management) rendszerét. A tettest, Vladimir Levin orosz számítástechnikust egy év múlva tartóztatták le a London–New York repülőjáraton, amikor már 10 millió dollár körül volt a huszonéves orosz ténykedése következtében a bank veszteséglistáját terhelő összeg.

2000 nyarán Ralph Dressel angol szoftverelemző felfedezett egy biztonsági rést, így hozzáfért az egyik legnagyobb amerikai internet banking szolgáltató, a Fiserv által vezetett mintegy 200 milliós bankszámlához, beleértve a PIN-változtatási és átutalási opciókat is. Két hónappal előtte a legnagyobb brit online bank, a Barclays kényszerült online szolgáltatásának átmeneti szüneteltetésére, miután ügyfelei a bejelentkezés után hozzáfértek más ügyfelek számlainformációihoz is. Hasonló eset történt 1999-ben az Egg és a Halifax online bankoknál. Ezek a biztonsági rések mindig valamilyen szoftverfrissítés „eredményeként" keletkeztek.

Magyarországon 2001 elején több mint 53 millió forintos kárt okozott egy bűnöző páros, amely internetes áruházat hozott létre, hogy fiktív vásárlások alapján pénzt csaljon ki a velük szerződésben álló banktól. Közölték a bankkal, hogy 214 személy vásárolt tőlük, és felszólították a pénzintézetet, hogy utalja át a vásárlók által a bankhoz befizetett összeget. Ez volt az első ilyen hazai eset. Előzőleg, 2000 őszén a Budapest Bank a Teleinvest névre hallgató internetes befektetési jegy kereskedelmi rendszerének leállítása mellett döntött, amikor visszaéléseket tapasztaltak a rendszerben.

rések szerint a vállalatok biztonsági kockázatainak túlnyomó hányadát kapun belül lehet megtalálni.)

A NetBoss és a biztonság

Az alábbiakban az Online Üzleti Informatika Rt NetBoss termékének biztonsági vonatkozásait mutatjuk be, érzékeltetve, hogy az említett általános biztonsági követelmények hogyan jelennek meg egy konkrét üzleti alkalmazásban. A magyar banki informatikai piacon meghatározó jelentőségű NetBoss biztonsági rendszere világszínvonalú, amit a független szakértői minősítések is megerősítenek.

A NetBoss rendszer külső és belső kommunikációja a következő biztonsági követelményekre támaszkodik:

a) Az internet felől.

— Csak HTTPS protokollal és csak a 443-as porton történhet kommunikáció.

— A NetBoss rendszer nem ad helyet a bank egyéb webszervereinek (nem kerülhet a rendszerbe alkalmazásidegen adat).

— Csak kliensoldali Java scriptek vannak az alkalmazásban.

— Nincs az alkalmazásban Active-X.

b) A belső kommunikáció oldaláról.

— A szerverek csak az előre megadott portokon és protokollok szerint kommunikálnak.

— A szerverek közötti összes kommunikáció naplózásra kerül.

Nyilvános csatornán

Az internet nyilvános csatornának számít, ezért a rajta továbbított adatokat a NetBoss védi a lehallgatás, az ellopás és a meghamisítás ellen. A NetBoss az SSL titkosítási protokollt használja 2048 bit kulcshosszúságú RSA és 128 bites DES kódolási algoritmussal, valamint a Global Server ID technológiával.

A módszer lényege a következő lépésekben foglalható össze.

1. Az ügyfél a NetBosstól kapott weboldalakat a számlavezető bank (SZB) privát kulcsával SSL protokoll szerint titkosítva kapja meg.

2. Az ügyfél szintén megkapja az SZB-től a bankhoz tartozó Global Server ID-t (GSID), ami azonosítja a SZB NetBoss rendszerének egyik szervergépét (gyakorlatilag magát a számlavezető bankot).

3. A GSID-ben a Verisign cég privát kulcsával titkosítva található a SZB publikus kulcsa; vagyis az a kulcs, amelynek segítségével az ügyfél böngészője a kapott weboldalakat dekódolni tudja. Mivel az SZB privát kulcsát csak az SZB ismeri, ezért az ügyfél által kért oldalt gyakorlatilag lehetetlen meghamisítani.

A lehallgatás vagy forgalomelterelés elleni védekezés részeként az ügyfél és az internet banking rendszer között kiépült adatkapcsolat integritását is védeni kell. A kapcsolat (session) védelme az alaptechnológiát jelentő Progress eszközök (Webspeed) szolgáltatásaira, valamint az időbélyeges regisztrálás (a cookie-k) használatára épül. A védelem része többek között a session automatikus lezárása, ha az ügyfél egy meghatározott időn túl inaktív marad.

Naplózott, proaktív, reaktív

Az állapotadatok gyűjtésének és naplózásának (log) lényege a rendszerben bekövetkezett események adatainak rögzítése. A NetBoss minden eseményt duplán naplóz, a napló másolata pedig a keletkezés helyéről átkerül egy központi gépre. Így ha egy támadás során a szerver saját naplóállományai esetleg megsérülnek, a központi adatokból rekonstruálhatók az események, másrészt jogvitában (ha az ügyfél tagadja bizonyos műveletek elvégzését) a naplóállományokkal bizonyítható a tranzakció megtörténte.

A proaktív biztonság a rendszerbe megelőzési céllal beépített védelmet jelenti.

A NetBoss a következő strukturális védelmi elemeket tartalmazza:

— Routeres csomagszűrés, tűzfal, hálózati szegmentálás.

— Fokozott koncentráció más (például banki számlavezető) rendszerekkel kialakított kommunikációs felületekre és a megengedhető kommunikációtípusokra (a TCP/IP portok száma, az alkalmazott protokoll milyensége stb.).

Szoftveres alkalmazási védelem:

— Jelszó-öregítési mechanizmus.

— Olvasási és írási jogok magas szintű menedzselése.

— Duálisan naplózott rendszeresemények.

— Az alkalmazott operációs rendszerek biztonságossá tétele („szekurizálása").

A reaktív biztonság a behatolási kísérletek valós időben történő figyelését és a megfelelő reagálást jelenti. A rendszer védekező reakciói közé soroljuk a támadási minták automatikus felismerését és a többi behatolásdetektáló módszert — feltételezve, hogy illetéktelen külső behatolóról van szó. A rendszerbe beépített behatolásfigyelő (intrusion

Egy kis mó(d)szertan

Billentyűzetfigyelés. A megtámadott gépen „lehallgatják" a billentyűleütéseket, rögzítik a kapott kódokat. Elsősorban felhasználói nevek és jelszavak ellopására használják.

Jelszókitalálás. A könnyű megjegyezhetőségre alapozott jelszavak kitalálására specializált módszer. Az erre készült programok használói leginkább a saját maguk által összeállított szótárral dolgoznak.

Jelszófeltörés. A titkosított állományok dekódolását nyers erő (brute force) alkalmazásával, algoritmikus módszerereket használó programok bevetésével megkísérlő támadás.

Véletlen tárcsázás. Egy bank összes telefonszámának vagy feltételezett számtartományának végigtárcsázása modem után kutatva. Megtalálása esetén a cracker ezen a behatolási felületen próbál továbblépni.

Szélhámoskodás. Gúnyos amerikai elnevezéssel „social engineering". A bank ügyfélszolgálatán vagy más emberi szálon történő megközelítés, például más nevében történő fellépés, jelszóváltoztatás kezdeményezése stb.

Trójai faló. A rendszerbe álcázással bejuttatott alkalmazás, amely hátsó kaput tud nyitni a kívülről történő behatoláshoz.

Újabban egyre gyakoribbak az olyan a visszaélések, amelyek inkább tekinthetők szabotázsnak, semmint betörésnek, ugyanis a rendszerbe való behatolás nem bizonyos adatok megszerzésére irányul, hanem „csak" a rendszer normális működésének lassítására vagy átmeneti megbénítására. Egy cég szolgáltatásképtelenné tétele (DoS, denial of service) ellen természetesen ugyanúgy védekezniük kell a biztonsági szakembereknek, annál is inkább, mert a behatolási módszerek általában azonosak, mint amilyeneket a lopási és csalási célzatú (magyarán pénzszerző) akciókban használnak.

detection) szoftver képes az előre definiált támadási minták észlelésére. Amennyiben a bemeneti jelfolyamban az előre definiált mintára illeszkedő támadási módot vesz észre, végrehajtja a hozzárendelt akciót. A támadási minták adatbázisa természetesen frissíthető, így a betörésvédelem naprakész felkészültségűvé tehető.

Azonosítás és jogosultság

A NetBoss kiemelkedő tulajdonsága, hogy többféle biztonsági szinten enged hozzáférést szolgáltatásaihoz. Attól függően, hogy milyen módon azonosította magát, az ügyfél eltérő — szűkebb vagy bővebb — szolgáltatáscsomaghoz férhet hozzá. A NetBoss rendszerben alkalmazott biztonsági szintek:

0. szint — nincs ügyfél-azonosítás.

1. szint — statikus jelszóval történik az azonosítás.

2. szint — dinamikus jelszóval történik az azonosítás.

3. szint — valamely hardvereszköz birtoklásához kötődik az azonosítás.

A rendszer lehetővé teszi, hogy a számlalekérdezési funkció például már az 1. szinten hozzáférhető legyen, de mondjuk az összeghatár nélküli átutalás már a 3. biztonsági szinten történő azonosításhoz legyen kötve. Nézzük meg, milyen eszközöket kínál a rendszer az egyes azonosítási elvek gyakorlati megvalósításához.

Dinamikus jelszóküldés

Dinamikus jelszóról beszélünk, ha az ügyfélnek minden bejelentkezésekor egy új, korábban nem használt jelszót kell használnia. A módszer régóta ismert, de alkalmazása kezdetben nehézkes és kockázatos volt, hiszen egy jelszóhalmazt előre kiosztottak neki papíron.

Az Online korszerűbb és kényelmesebb megoldást alkalmaz: a NetBoss SMS üzenetben küldi ki az ügyfélnek az egyszer használatos jelszót, a jelszógenerálás pedig fizikai véletlen számokon alapul, ezért algoritmikusan visszafejthetetlen.

Azonosítás hardvereszközzel

A dinamikus jelszavak használata a hagyományoshoz képest komoly biztonsági előrelépés, de a publikus csatornákon történő továbbítás kockázatot is rejt magában. A kockázat jelszógeneráló eszközök (one-time-pad vagy hardver-token) alkalmazásával kiküszöbölhető. Ezek az zsebszámológépre emlékeztető eszközök a megfelelő PIN kód megadása után dinamikus jelszót állítanak elő, amelyet a NetBoss bekér az ügyféltől, hogy azt az erre a feladatra használt szerver azonosíthassa. A jelszó előállítása olyan algoritmussal történik, hogy a szerver egyértelműen megállapíthassa a jelszó hitelességét, ugyanakkor a jelszó egyedi, és csak egyszer használatos. A megoldás előnye, hogy növeli a biztonságot, eközben mégsem csökkenti az ügyfél mobilitását, hiszen nem szükséges hozzá további periféria vagy szoftver.

A NetBoss természetesen támogatja a PKI alapú ügyfél-azonosító megoldásokat, így az intelligens vagy okos kártyák használatát is (smart card). Számos előnyös tulajdonságuk mellett az intelligens kártyák hátránya, hogy kártyaolvasó kell hozzájuk, ami egyelőre nem standard része a PC konfigurációnak. Alternatív megoldásként az Online támogatja az e-token azonosító használatát. Ez a kulcstartóra emlékeztető eszköz nem igényel olvasót, egyszerűen rácsatlakoztatható a PC USB portjára.

Kovács Balázs
balazs.kovacs@online.hu

Csekk vagy bankkártya

Hamis aláírás esetén a banknak nincs joga a csekktulajdonos számláját megterhelni. Amennyiben ez mégis megtörténik, köteles az összeget a számlára visszahelyezni. Nagy összegű tranzakciók esetén nemcsak tüzetesen összevetik az aláírást az aláírásmintával, hanem gyakran telefonon is jóváhagyatják az ügyféllel a tranzakciót. A hamisított csekkek akkor okoznak nagyobb problémát, ha például egy kereskedő a banknál történő beváltáskor szembesül a ténnyel: hamisított csekket fogadott el. Ennek kivédésére kisebb összegű vásárlásokra a 60-as években bevezették a csekk hitelességét garantáló kártyák használatát, így a kereskedőnek van lehetősége az ellenőrzésre.

Bár az aláírás ma is a leggyakrabban használt azonosító eszköz, önmagában nem igazán megbízható, a professzionális írásszakértők is 6,5%-os tévesztési aránnyal dolgoznak, laikusok esetében pedig a 40%-ot is eléri azok aránya, akik rosszul ítélik meg egy aláírás valódiságát.

Az Egyesült Államokban az 50-es, Európában a 60-as években jelentek meg a bankkártyák, de jelentős piaci felfutásuk a 80-as évekre esik. A csekkforgalomtól biztonsági szempontból annyiban különböznek, hogy az aláírás szerepe korlátozottabb, továbbá előfordul, hogy az ügyfél akkor is visel bizonyos felelősséget, ha a kártya kikerül a rendelkezése alól. Ha azonban a kártyatulajdonos jelen van a tranzakciónál, és a kereskedő a tőle elvárható gondossággal jár el, akkor a bank a kockázatviselő fél.

Ha egy kártyás vásárlásnál az ügyfél személyesen nincs jelen, a visszaélés kockázata nagyobb, hiszen elegendő a kártya típusát, számát, lejáratának dátumát és a tulajdonos nevét ismerni. Ha tehát a kártyatulajdonos kifogásol egy tranzakciót, és nincs bizonyíték arra, hogy az árut kiszállították került, vagy a szolgáltatást igénybe vette, a bank nem terhelheti meg az ügyfél számláját a tranzakcióval. Ebben az esetben a kereskedő oldalán van a nagyobb kockázat, hisz ilyenkor a bank egy „chargeback" eljárás keretében visszaköveteli a vásárlás ellenértékét, a kereskedő viszont már nem tudja másra áthárítani a kárt.

Ezek a kockázatok az elektronikus kereskedelem megjelenésével is fennmaradtak, hovatovább annak jelentős gátló tényezőjévé váltak. Web alapú hozzáférési csatorna titkosítása (TLS, SSL) esetén a kártyaadatok ugyan kódoltan kerülnek továbbításra, de ez a megoldás sem küszöböli ki a kártyaadatok széles körű hozzáférhetőségéből eredő problémát, valamint nem nyújt módot a kártyainformációt szolgáltató személynek és a kártya jogos birtokosának ellenőrzésére, így a kereskedő kockázata továbbra is fennáll.

A Visa és Mastercard által életre hívott SET (Secure Electronic Transactions) szabvány lehetővé teszi, hogy a kereskedő kérésére a bank azonnal megvizsgálja a kártyatulajdonos jogosultságait, és jóváhagyja a tranzakciót. Az új szabvány azonban körülményessége és költségessége miatt nem terjedt el, és az is valószínűsíthető, hogy minden olyan technológia, amely a kockázatviselést a kereskedő vagy a bank oldaláról a kártyatulajdonos felé tolja el, komoly ellenállásba ütközik, és nem erősíti az elektronikus kereskedelembe vetett bizalmat.

Most már digitálisan írunk alá?

Az elektronikus dokumentumok hitelessége

A nyílt hálózatok (mint például az internet) felépítéséből adódóan az üzenetek különböző szolgáltatók által kontrollált rendszereken haladnak keresztül, így fennáll annak a lehetősége, hogy az üzenetekhez nem csak a címzett fér hozzá. Kézenfekvő a kérdés, hogy létezik-e olyan megoldás, amely biztosítja a küldő és a fogadó egyértelmű azonosítását, az eredeti üzenet változatlanságát, és azt, hogy mások ne férjenek hozzá az üzenet tartalmához.

Az elektronikus dokumentumok titkosságára és megbízható azonosítására a legmodernebb mai technikai megoldás az aszimmetrikus, más néven nyilvános kulcsú titkosítás, amelynek legelterjedtebb változata az RSA. Az eljárás előnye, hogy a küldő és a fogadó félnek nem kell semmilyen titkos jelszót, kódot, kulcsot cserélnie egymással, az adatforgalom mégis teljesen biztonságosan lebonyolítható. A kommunikációban részt vevő felek mindegyike rendelkezik egy kulcspárral, ennek egyik tagját titkos (privát, magán-), míg a másikat nyilvános (publikus) kulcsnak nevezzük.

A kulcspár titkos részét csak annak tulajdonosa ismeri, míg a publikus kulcs mindenki számára elérhető. Az eljárás lényege, hogy a kulcspár egyik tagjával végzett művelet (kódolás) a másik taggal, a kódoló kulcs párjával visszafejthető (dekódolás). Az eljárás biztonságát pedig az adja, hogy az egyik kulcs ismeretében a másikhoz nem tudunk eljutni.

Titkosítás

Titkosítási műveletnek nevezzük azt, amikor a sokak által ismert publikus kulcsot használjuk kódolásra, a dekódolása pedig egyedül a megfelelő magánkulcsot birtokló személy képes.

Egy biztonságos üzenetváltás a következő módon zajlik Adél és Béla között. Adél és Béla is rendelkezik egy kulcspárral. Adél a Bélához tartozó publikus kulcs segítségével titkosítja az üzenetet, amelyet később csak Béla tud visszafejteni saját titkos kulcsa segítségével. A kulcspár használata lehetővé teszi, hogy az elektronikus üzeneteket (elektronikus leveleket, csatolt dokumentumokat, weboldalakat, megrendeléseket, pénzügyi utasításokat, számlalekérdezéseket) csak az üzenet címzettje tudja dekódolni.

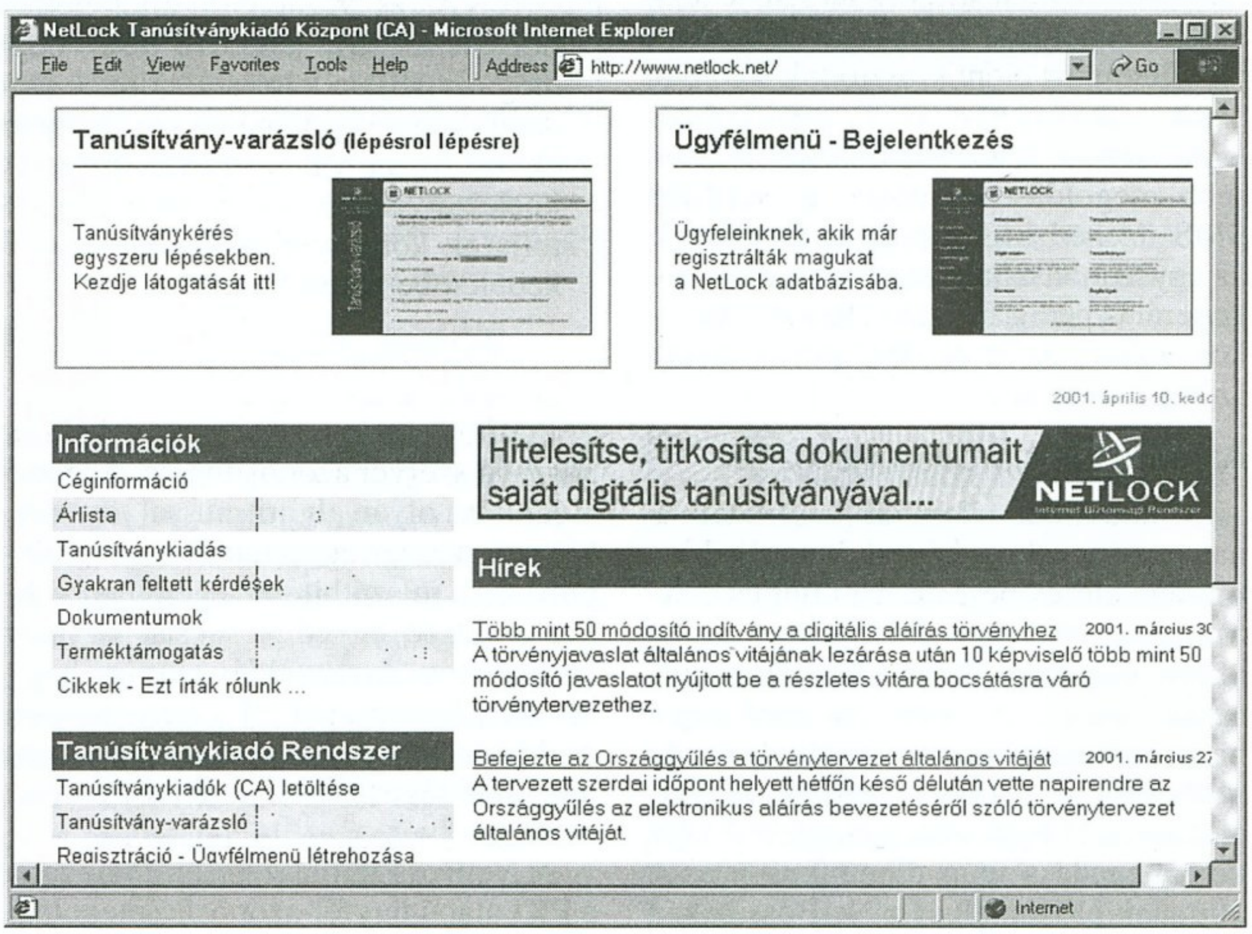
Digitális aláírás

Amikor a fenti eljárás fordítottjaként valaki az egy példányban meglévő magánkulccsal kódol valamit, majd ezt a nyilvános kulccsal sokan ellenőrizni tudják, ezt az eljárást nevezzük digitális aláírásnak. A digitális aláírás a hagyományos papíron történő aláíráshoz hasonlóan, de számítógépes környezetben teszi ellenőrizhetővé az adatok hitelességét, azaz a küldött üzenetek feladói pontosan azonosíthatók, és az üzeneteket nem lehet letagadni, tehát bizonyító erővel bírnak. A digitális aláírás rendelkezik a hagyományos aláírás összes fontos tulajdonságával: az aláírást csak az aláíró tudja előállítani, és mindenki könnyen ellenőrizheti, hogy tényleg a feltételezett aláíró készítette-e.

Az elektronikus aláírásnak van egy fontos és rendkívül hasznos tulajdonsága, mégpedig az, hogy a kézjeggyel ellentétben nem az üzenet hordozóanyagához tartozik, hanem magához az üzenethez. Ha két eltérő üzenetet látunk el kézjegyünkkel, aláírásunk ugyanaz marad, míg ha két eltérő üzenetet látunk el digitális aláírással, a két digitális aláírás különböző lesz.

A hitelesítés

A titkosítás és a digitális aláírás teljesen független két művelet. Bár ugyanazt a technológiát alkalmazzák, a két eljárás bármilyen kombinációban, külön-külön alkalmazható. Amennyiben a titkosítást és a hitelesítést egyszerre használják, úgy a fogadó oldalon a kulcsok párjait alkalmazva az eredeti szöveg is visszaállítható, az aláírás is ellenőrizhető. Ahhoz, hogy a technológiát a hétköznapi életben felhasználhassuk, elektronikus tanúsítványokra van szükség. Ezen tanúsítványok megbízható harmadik személyek, hitelesítő szervezetek (certification authority) által kerülnek kibocsátásra. A hitelesítési



szolgáltató szerepe az, hogy tanúsítsa egy felhasználó (személy, szervezet, vagy akár egy számítógép) és az általa használt titkos kulcs összetartozását. A felhasználót annak neve, internetes címe vagy más adatai alapján, a titkos kulcsot pedig a hozzá tartozó nyilvános párjával tudjuk egyértelműen azonosítani.

A szolgáltató rendszerint együttműködik regisztrációs szervezetekkel, amelyek feladata az, hogy ellenőrizzék az ügyféladatok hitelességét, valamint betartassák a különböző eljárási rendeket. A hitelesítő a megtörtént ellenőrzésekre támaszkodva bocsátja ki tanúsítványait, amelyeket saját digitális aláírásával hitelesít, azaz a kiadó megerősíti a tanúsítvány tulajdonosát azonosító adatoknak és a tulajdonos nyilvános kulcsának összetartozását.

Magyarországon a NetLock

Magyarország egyetlen nyilvános szolgáltatásokat végző tanúsítványkiadója a NetLock Kft, amely 3 osztályban (C, B, A) és 4 típusban (személyes, névjegykártyás, szervezeti, szerver) bocsátja ki elektronikus tanúsítványait. Ezek ára havonta 400-tól 2000 Ft-ig terjed. Az egyes osztályok nem a tanúsítványok technikai megbízhatóságára, hanem arra az ellenőrzési folyamatra utalnak, amelyen az ügyfeleknek keresztül kell menniük egy-egy tanúsítvány kibocsátása előtt. Egy tanúsítvány kibocsátásához C osztályban elegendő a megfelelő dokumentumok másolata, B osztályban a kibocsátás személyes megjelenéshez, míg A osztályban közjegyző által hitelesített dokumentumokhoz kötött. Az egyes entitások azonosítását a céghez kapcsolódó regisztrációs hatóságok segítik, amelyek közül említésre méltó a Magyar Posta 3200 hivatalával, valamint a magyar közjegyzői irodák rendszere.

Minden egyes kibocsátott tanúsítványt az Axa-Colonia termékfelelősség-biztosítása védi, amely 50 ezertől 5 millió forintig terjedően téríti meg a tanúsítványokkal okozott esetleges károkat. A céget 1999-ben a Microsoft hitelesítési szolgáltatóként elismerte, és nyilvános kulcsait termékeiben elhelyezte. Ez lehetővé tette, hogy az egyes szoftverek a NetLock Kft által kibocsátott tanúsítványokat — amelyek száma 80 ezerre tehető — automatikusan felismerjék, használják. Ezáltal a NetLock Kft az egész világon elismertté és elfogadottá vált.

A cég konzulensként részt vett Magyarországon az elfogadás előtt álló elektronikus aláírási törvény előkészítésében. A törvény feladata, hogy megteremtse az elektronikus aláírás jogi hátterét, valamint szabályozza azon szervezetek megalakulását és működését, amelyek a jövőben tanúsítványokat bocsátanak ki. A törvény meg fog különböztetni úgynevezett minősített (HIF minősítéshez kötött) és minősítéssel nem rendelkező hitelesítő szervezeteket és tanúsítványokat. A minősített tanúsítványokkal aláírt dokumentumok teljes bizonyító erejű magánokiratnak minősülnek majd, azaz megfelelnek a két tanú aláírásával ellátott hagyományos iratnak. Azon szolgáltatók is kibocsáthatnak tanúsítványokat, amelyek működésükhöz nem szerzik meg a HIF minősítését, ám ezek jogi ereje gyengébb. Ezen tanúsítványok használata esetén a bíróság egyedileg mérlegel minden olyan elektronikusan aláírt dokumentumot, amelynek hitelessége megkérdőjeleződik.

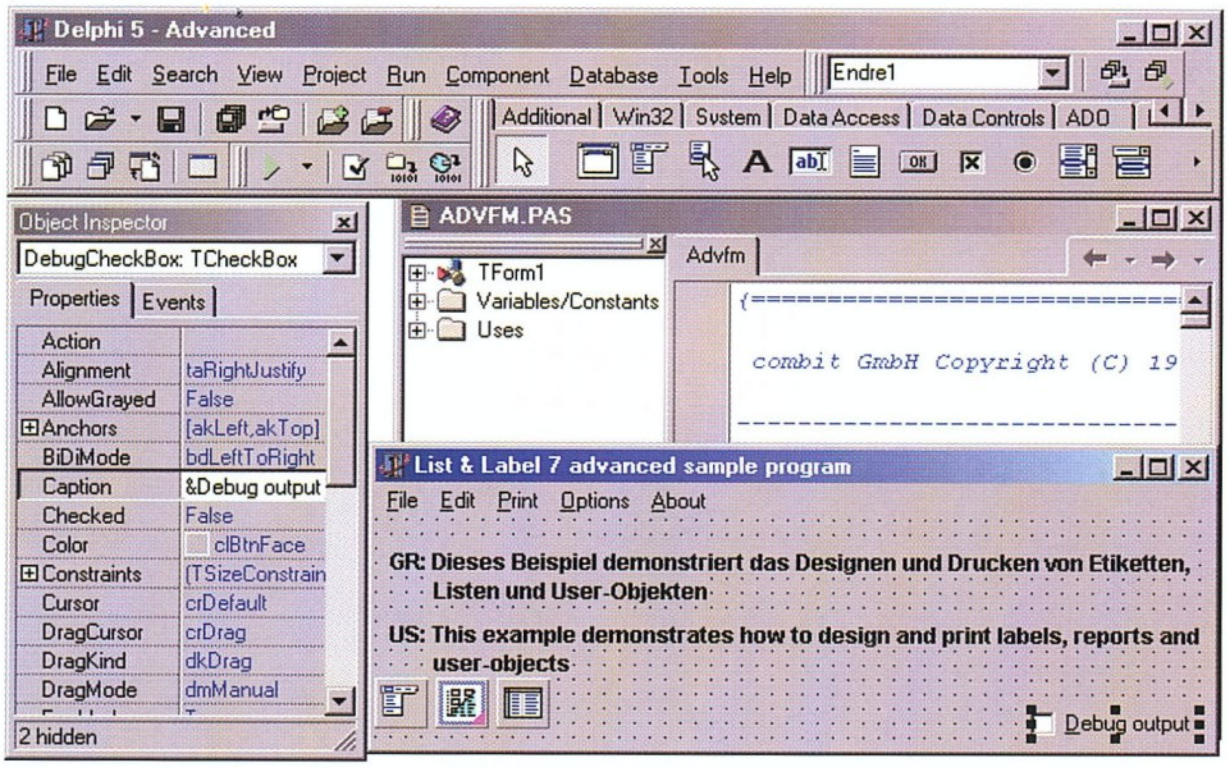
Berkes Tibor
berkes_t@netlock.net

Listák és címkék komponensből

List & Label 7.0 — jelentéskészítéstől a körlevelekig

Adatbázisokból adatok lehívására gyakran alkalmazott mód a listák, címjegyzékek, címkék kinyomtatása. Ahhoz azonban, hogy ezeket saját ízlésünknek megfelelően készítsük el, nem okvetlenül van szükségünk „nehézfegyverekre". Egy jól programozható célalkalmazás sokszor alkalmasabb megoldásokat kínál. A Combit cég által (http://www.combit.de) kifejlesztett List & Label program, pontosabban komponenscsomag számos programozási környezetben használható. Windows platformon a bináris formában telepített komponens kívülről elérhető függvényeit a Registry közvetítésével más alkalmazások is meghívhatják, függetlenül a fejlesztési nyelvtől. Felhasználásukhoz programozói oldalról nézve csak a külső komponensekkel végzett munka támogatására alkalmas fejlesztőrendszerre és a függvények dokumentációjára van szükség.

A fejlesztőkörnyezettel ma már általában nincs is gond, viszont a programozóknak nem mindig erősségük a dokumentálás. Kényelmesebb is közvetlenül felhasználható anyagokat kapni, például fejlesztőcsomagok vagy ActiveX készletek formájában. A List & Label legújabb, 7-es verziója a kettő ötvözetének tekinthető: a telepítése után kapott rendszer nem tipikus SDK, közel 17 MB a terjedelme, és sokrétűsége révén más fejlesztőeszközökkel is használható (C++ Builder, Centura for Windows, Watcom C, az XBase stb.). A forrásfájlokon kívül példaprogramok is segítik a munkát, de csak a gyakoribb fejlesztőeszközökre, amilyen a Delphi, a Visual C++ és Visual Basic. Magyarországon a Ker-Soft (www.kersoft.hu) forgalmazza.






symantec.
Symantec and the Symantec logo are U.S. registered trademarks. ©2001 Symantec Corporation. All Rights Reserved.
Behatolás megelőzés a Symantectől
A hackerek, szabotőrök és cyber bűnözők mostantól új hobby után nézhetnek! A Symantec Enterprise Security távol tartja őket rendszerétől! Az új technológia biztosítja teljes hálózatát: a szervereket, a távoli felhasználókat, a webes alkalmazásokat és az asztali gépeket egyaránt. A Symantec a szolgáltatást és a tudást összehangolva teszi biztonságossá számítástechnikai környezetét. További információkat találhat a www.symantec.hu vállalati megoldások oldalán.


A szélessávú internetszolgáltató...
TVNET Számítástechnikai Kft.
1135 Budapest, Csata u. 8.
Tel.: 236-6250 Fax: 236-6251
info@tvnet.hu
www.tvnet.hu
Kábeltévén - a leggyorsabban
Bérelt vonalon - a legmegbízhatóbban
Mikrohullámon - a legrugalmasabban
ADSL-en - a leghatékonyabban
TVNET új
www.tvnet.hu

A vírusírók „szponzora”

Office XP és a makróvédelem

Minden új Microsoft Office verzió kihívást jelent a vírusírók és a vírusok elleni védekezésre szakosodott szakemberek számára. Az Office XP (leánykori nevein Office 2002, OX...) Béta 2 változatát tesztelgetve a szokásos makróbiztonsági párbeszédablakon megláttam a „Trust access to Visual Basic Project” feliratot. Bár szerényen az ablak legalján helyezkedik el, és a szöveg sem sokatmondó, amit reprezentál, az valójában nagyon fontos.

A makróvírusok életképességének feltétele, hogy hozzáférjenek egymástól nagyon távol álló két objektummodellhez. Az egyik az alkalmazás objektummodellje, amely lehetővé teszi a vírus aktivizálódásához szükséges folyamatokat (például a dokumentum megnyitásakor elinduló Document_Open eljárást), a másik a VBE (Visual Basic Editor) objektummodell, amely a víruskód manipulálásához és más dokumentumokba való beillesztéséhez szükséges eljárásokat adja. Hogy a két objektummodell között milyen fokú átjárást tesznek lehetővé, azt a szoftverfejlesztő cégek maguk dönthetik el, amikor alkalmazásaikba integrálják a VBA-t (Visual Basic for Applications).

A VBA-t használó egyes irodai alkalmazáscsomagok, így elsősorban a Microsoft Office, könnyű hozzáférést biztosítanak mindkét objektumtípushoz. Más alkalmazások, mint például a WordPerfect, teljesen leblokkolják a hozzáférést mindenféle VBE-objektumhoz. Mivel a jelenleg ismert WordPerfect vírusok száma közel van a nullához, könnyen kitalálható, hogy biztonsági szempontból melyik megoldás a jobb.

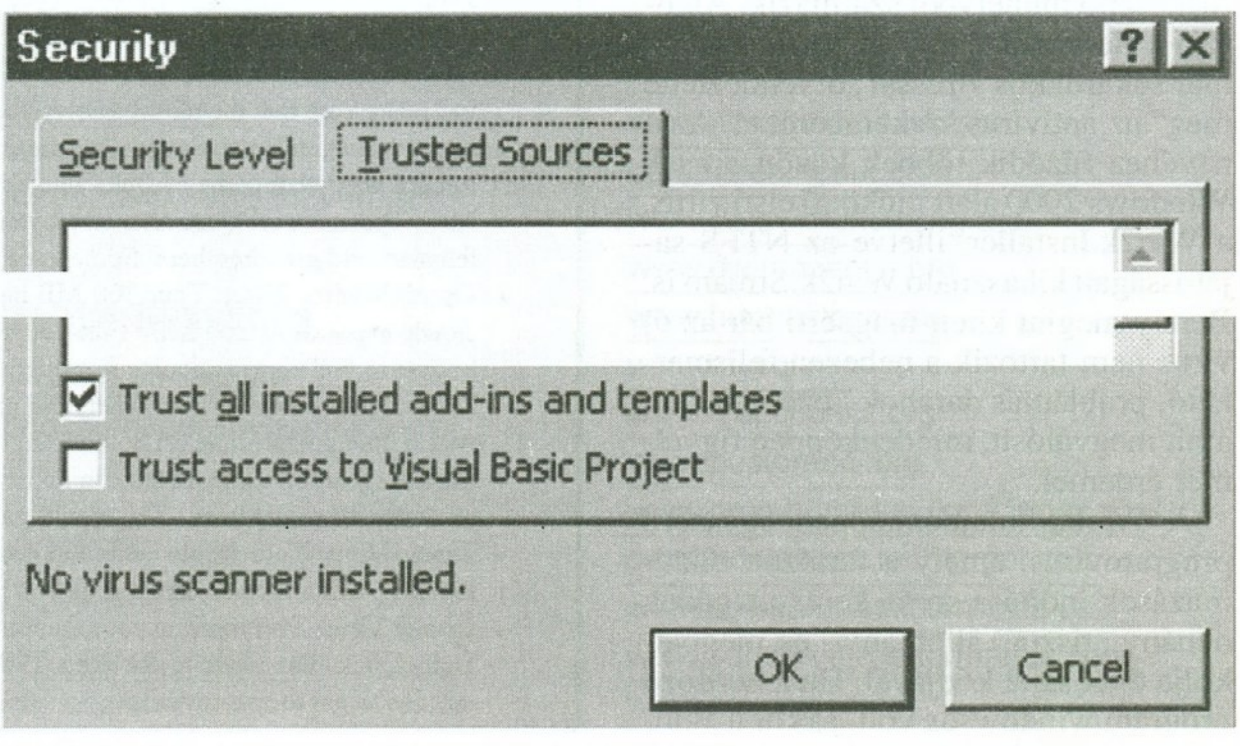
Az Office XP makróbiztonsági ablakában az alapértelmezés a Visual Basic projektre utaló beállítás kikapcsolt értéke. Ha ilyenkor egy tipikus Word makróvírust akarunk elindítani, annak aktivizálódása minden látható ok nélkül elmarad. A további kutakodás érdekében a vírust most már a Visual Basic szerkesztőben elindítva egy másik hibaüzenetet kapunk, amely közli, hogy a Visual Basic projekt elérése nincs engedélyezve. Ekkor kezd érdekessé válni a dolog. Vajon mit talált ki a Microsoft? Talán nem engedi tovább érvényesülni kedvenc makróvírusainkat, hiszen túlnyomó részüknek létszükséglet a VBE-objektumok hozzáférhetősége. Nagyon kis hányaduk operál más módszerekkel, és csak ők maradnának életképesek Office XP alatt is.

Azután elkezdtem gondolkodni. Ha az objektummodellek szeparálása egy kapcsolóval történik, akkor azt nyilván ki-be lehet kapcsolni. Amit ki-be lehet kapcsolni, annak pillanatnyi beállítási értékét valahol tárolni kell. Ha adatot kell tárolni, akkor azt a Microsoft a regisztrációs adatbázisban tárolja. Rövid kutakodás után sikerült is megtalálnom a bejegyzést, méghozzá elég szembeötlő helyen. Hát eddig terjed az Office XP-ben bevezetett biztonság. Ilyen keresgélést egy közepesen rátermett vírusíró is végre tud hajtani, a regisztrácós adatbázisban való turkálás pedig manapság rutinfeladat egy makróvírusnak.

Annyi haszna persze lehetne ennek a javításnak, hogy legalább a már ismert makróvírusok ne működhessenek Office XP alatt. A VB-opciót azonban ki lehet kapcsolni, tehát a minimális cél elérése sem reális. Az újonnan megírandó Office XP makróvírusok pedig már rutinszerűen tartalmazni fogják a vírusvédelem hatástalanítására szolgáló néhány programsort. Ugyanez történt az Office 97 Service Release 1-ben bevezetett extra vírusvédelemmel. Most sem lehet másra számítani, sőt a hírek szerint már meg is jelent az első Office XP-re írt Word makróvírus, ami teljesen logikus is, hiszen az utóbbi években az Office új változatainak boltba kerülése előtt kb. másfél hónappal általában felbukkantak a rájuk írt első makróvírusok.

Ami leginkább bosszantó az egészben: most tényleg ki lehetett volna húzni a makróvírusok méregfogát. Csak egy kicsit kellett volna bátornak lenni, és különösebb teketória nélkül megtiltani mindenféle hozzáférést a VBE-objektumokhoz. Kevesen vették volna észre a megszorítást, talán csak az a néhány fejlesztő, aki VBA alapú kódrendszerező alkalmazásokat használ (például a Visual Source Safe ilyen), illetve aki programkódot változtató adaptív alkalamazásokat fejleszt. Nem ez az első alkalom (és félek, hogy nem is az utolsó), amikor a Microsoft a fejlesztők kedvében akar járni, és inkább figyelmen kívül hagyja a többség érdekét, a biztonsági szempontokat.

Szappanos Gábor
gszappanos@vbuster.hu



Platformfüggetlen vírusok

Winux: „What do you want to infect today?”

A vírusok eddig általában csak egy-egy platformot céloztak meg. A túlnyomó többséget kitevő windowsos vírusokat éppúgy az erős platformfüggőség jellemezte, mint a Linux, PalmOS, BeOS, Mac vagy OS/2 rendszerre jóval kisebb számban megírt vírusokat. Most azonban felbukkant egy olyan programvírus, amely egyszerre ténykedik a két domináns 32 bites környezetben: megfertőzi a windowsos és az Intel alapú linuxos állományokat is.

A vírusok platformfüggetlensége nem egészen új keletű. Vannak olyan vírusok, amelyek megfertőznek Word dokumentumokat és végrehajtható állományokat is. Ilyen például a Cocaine, amely fertőzött dokumentumban vagy fertőzött programmal is érkezhet, és mindkét esetben képes elszaporodni a számítógépen. Az Esperanto nevű vírus pedig nemcsak a Windows, hanem a Macintosh rendszer bináris állományait is áldozatul ejti.

Eddig azonban nem nagyon kellett tartani attól, hogy windowsos gépünket megtámadja egy merőben más platformra készült vírus. Annak esélye például, hogy az Esperanto egy fertőzött Macintosh gépről Windowst futtató gépre kerüljön át, nagyon csekély volt, mert a Windows és a Mac használói általában nem csereberélik egymás közt a végrehajtható állományokat. A most felbukkant Winux vírussal már egészen más a helyzet.

A Winux egy csehországi, brnói víruskészítő műhelyből származik. Alkotója a Benny álnéven ismert figura, aki már sok trükkös vírussal „örvendeztette meg” az antivírus szakembereket. Az ő nevéhez fűződik többek között a csak Windows 2000 alatt működő első vírus, a Win2k.Installer, illetve az NTFS sajátosságait kihasználó Win2k.Stream is. Benny megint kitett magáért: bár az új vírus nem tartozik a nehezen felismerhető, problémás darabok közé, az ötlet, amit megvalósít, mindenképpen figyelmet érdemel.

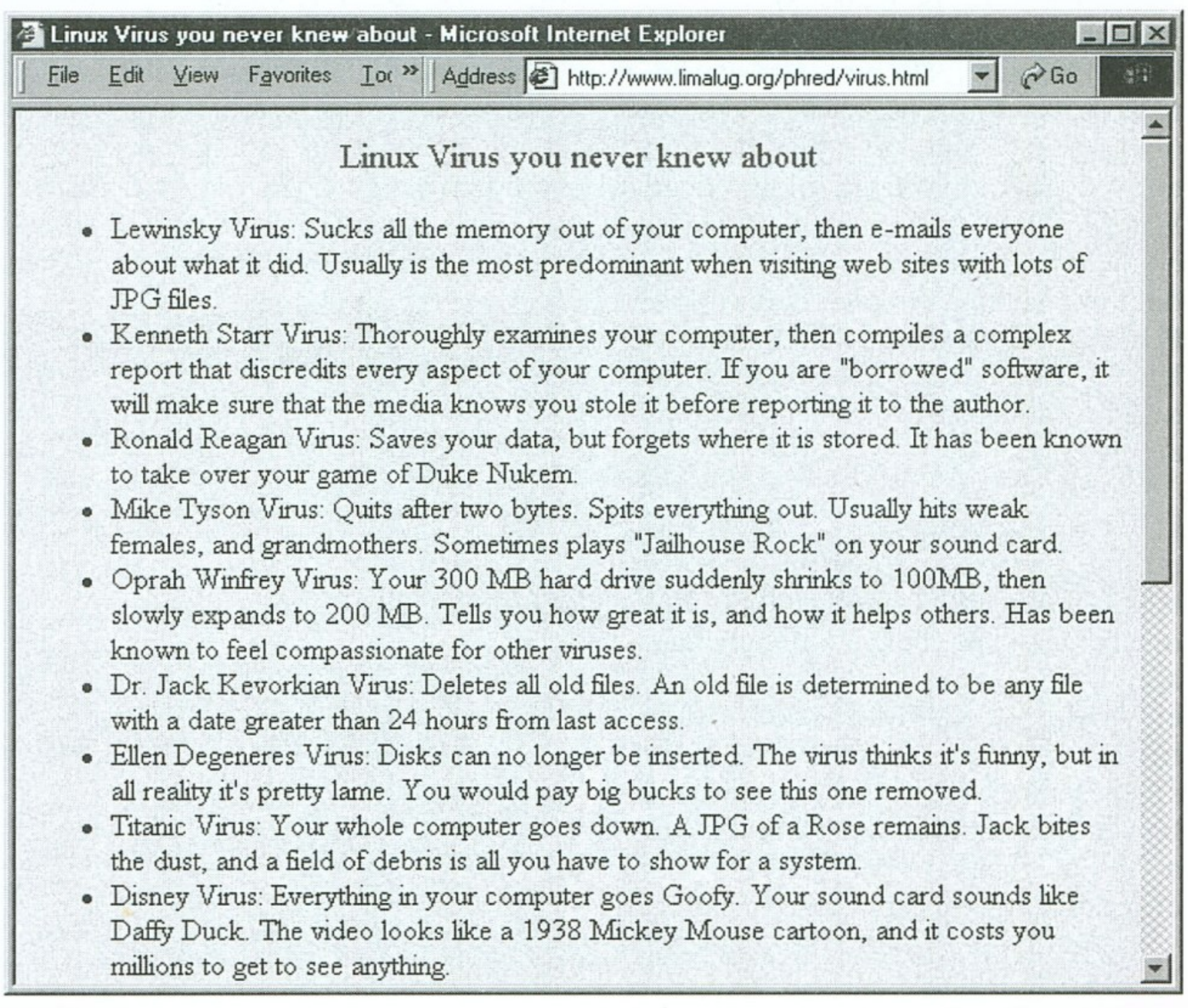
A Winux tulajdonképpen egyszerű programvírus, amely a fertőzött alkalmazások indítása során keres magának néhány „tiszta” állományt, és megpatkolja őket saját kódjával. Ha a hordozó program windowsos volt, akkor a Windows alá megírt kód kap vezérlést, és a vírusoknál szokványos módon, a Windows API-k használatával keres és fertőz fájlokat. Ha viszont linuxos alkalmazással kerül be a vírus a rendszerbe, akkor a Linuxra készült kódrész fog lefutni, és a linuxos vírusok gyakorlatának megfelelően a kernel interfész segítségével operál. A fertőzött program lefuttatásakor a vírus áldozatok után kutatva átfésüli az aktuális könyvtárat és annak szülő könyvtárát. Akár Windows, akár Linux környezetben futtatható alkalmazásokat talál, meg tudja őket fertőzni.

A gyakorlatban ez azt is jelenti, hogy ha egy gépen Linux és Windows is van telepítve, és a vírus „látja” a másik operációs rendszer partícióját, akkor gond nélkül megfertőzheti az azon található állományokat. A dolog mindkét irányban tökéletesen működik. Sőt, a vírus még akkor is képes terjedni, ha egy fertőzött windowsos alkalmazást Linux alatt, Wine emulátor segítségével futtatnak.

A Winux nem végez semmilyen pusztítást, és más módon sem tudatja a felhasználóval a jelenlétét. Miután megfertőzött a rendszerben néhány állományt, szépen engedi futni a vírust hordozó alkalmazást is, tehát jelenléte az állományok méretnövekedését leszámítva nem tűnik fel.

Azt régóta tudjuk, hogy a Linux nem vírusmentes platform. Még a gyorsan elhíresült IloveYou vírusnak is van unixos változata. És minél jobban terjed a Linux operációs rendszer, annál inkább áldozata lesz a vírusoknak és egyéb rosszindulatú programoknak. A jelenlegi néhány tucatról nagyon gyorsan néhány százra nőhet a Linux alatt (is) életképes vírusok, trójai programok, férgek száma.

Demeter Zoltán
zdemeter@vbuster.hu



Linux Virus you never knew about

- Lewinsky Virus: Sucks all the memory out of your computer, then e-mails everyone about what it did. Usually is the most predominant when visiting web sites with lots of JPG files.
- Kenneth Starr Virus: Thoroughly examines your computer, then compiles a complex report that discredits every aspect of your computer. If you are "borrowed" software, it will make sure that the media knows you stole it before reporting it to the author.
- Ronald Reagan Virus: Saves your data, but forgets where it is stored. It has been known to take over your game of Duke Nukem.
- Mike Tyson Virus: Quits after two bytes. Spits everything out. Usually hits weak females, and grandmothers. Sometimes plays "Jailhouse Rock" on your sound card.
- Oprah Winfrey Virus: Your 300 MB hard drive suddenly shrinks to 100MB, then slowly expands to 200 MB. Tells you how great it is, and how it helps others. Has been known to feel compassionate for other viruses.
- Dr. Jack Kevorkian Virus: Deletes all old files. An old file is determined to be any file with a date greater than 24 hours from last access.
- Ellen Degeneres Virus: Disks can no longer be inserted. The virus thinks it's funny, but in all reality it's pretty lame. You would pay big bucks to see this one removed.
- Titanic Virus: Your whole computer goes down. A JPG of a Rose remains. Jack bites the dust, and a field of debris is all you have to show for a system.
- Disney Virus: Everything in your computer goes Goofy. Your sound card sounds like Daffy Duck. The video looks like a 1938 Mickey Mouse cartoon, and it costs you millions to get to see anything.

Apróhirdetés

aula.online.hu/wp/main.htm
www.egyxegy.hu
www.expressz.hu
www.externet.hu/kereskinal
www.interapro.hu
mobil.naplopok.hu/aprohirdetes
www.stop.hu/apronet
195.228.240.145/apro

Álláshirdetés

www.allasajanlat.hu
www.allascentrum.hu
www.allaskozvetites.hu
www.allaspont.hu
www.cvonline.hu
www.humanlabor.hu
www.job.hu
www.jobline.hu
www.jobpilot.hu
www.jobscout24.hu/Neptun
www.job4smarts.com
www.karrier.hu
www.karrierexpressz.hu
www.solana.hu
www.tavmunkainfo.hu

Általános webhírlap

www.comedia.hu
www.curier.hu
www.eol.hu
www.internetto.hu
www.index.hu
www.korridor.hu
www.megaport.hu
www.mindenkinet.hu
www.mti.hu
www.netkapu.hu
www.netlap.hu
www.origo.matav.hu
www.stop.hu
www.vianovo.hu

Számítástechnikai webhírlap

www.cdgrab.hu
www.hwhunpage.com
www.hwsw.hu
www.pontjo.hu
www.prim-online.com
www.supergamez.hu
www.szamitastechnika.hu
www.szamitogep.hu
www.terminal.hu
www.wap.hu

Gazdasági webhírlap

www.ebroker.hu
www.eco.hu
www.fn.hu
www.napi.hu
www.portfolio.hu
www.quaestor.hu

Tematikus informatikai honlap

www.bsd.hu
www.driver.hu
www.extra.hu/verebics
www.gnome.hu
www.ini.hu
www.isz.hu
www.kde.hu
www.linux.hu
mobil.hix.com
www.mobilvilag.hu
www.nexus.hu/netjog
plusabit.tvnet.hu
www.tesztelo.hu
www.vbuster.hu
wigwam.sztaki.hu

Számítástechnikai boltlista

alag3.mfa.kfki.hu/dcsabas
/hardware/ceglist.htm
www.depo.hu
pons.sote.hu/~patherz/cegek.html

Szoftverletöltés

www.prim.hu/letoltes
tucows.euroweb.hu
tukor.fabricius.hu
www.xlr8.hu
195.228.240.145/szoftverbazis

Szoftverhonosítás

www.lme.hu/forditas
honositomuhely.hypermart.net

Könyvtár

www.mek.iif.hu
www.neumann-haz.hu
www.oszk.hu

Oktatás

www.nyelvkalauz.hu

Szótár

www.cab.u-szeged.hu/cgi-bin/szotarK
www.cab.u-szeged.hu/cgi-bin/szotarG
www.inf.elte.hu/~chaos/latin
szotar.sztaki.hu/angol-magyar

Keresőrendszer

altavizsla.origo.hu/katalogus
www.goliat.hu
www.heureka.hu
www.honlaptar.hu
www.hudir.hu
www.kapu.hu

Kezdőoldal, linkgyűjtemény

e1.hu
www.hirek.hu
www.optimax.hu/linkek/index_b.htm
www.webmutato.hu
www.start-hu.com
startlap.com
uzlet.lap.hu

Szakmai szervezet

www.ivsz.hu
www.njszt.iif.hu

Toplista

www.hungariantop1000.com
www.tipptop.com
top100.isys.hu
www.yahun.hu

A hónap témájához

www.altaplana.com/olap
databases.about.com
www.datawarehousing.org
www.dwinfocenter.org
www.hughes.com.au
www.mysql.com
www.olapcouncil.org
www.postgresql.org
www.prog.hu/cikkek/html/index.php3
searchdatabase.techtarget.com
www.software.ibm.com/data/db2
www.sql-server-performance.com
www.sql.org

Shakespeare vagy Bacon?

Az Új Alaplap 2001. februári számának 59. oldalán Faklen Pál írásában szerepelt egy megjegyzés, hogy „William Shakespeare neve mögött Francis Bacon alkotta meg a világirodalom remekműveit". Ha lett volna mellette valami olyan kitétel, hogy „egyes feltételezések szerint", akkor nem is reagáltam volna rá, de határozott állításként ez így nem fogadható el.

Való igaz, hogy nem maradt fenn Shakespeare 37 színművének eredeti kézirata, sőt alkotójukról sincs túl sok hiteles tárgyi emlék. Ahhoz azonban elegendő, hogy segítségükkel, de főleg a kortársak emlékei és visszaemlékezései alapján kirajzolódjon előttünk ennek az emberi jellemeket és a színészet mesterségét mélyen ismerő színésznek, színigazgatónak, rendezőnek és kivételesen termékeny írónak a személye.

Francis Bacon rendkívül széles látókörű, sokoldalúan művelt arisztokrata főúr volt, aki apjának, a lordpecsétőrnek a gondoskodásával már ifjú korában megismerkedett a tudományok alapjaival, sőt a korabeli Európa földrajzi és politikai viszonyaival is. Évekig az angol követ kíséretének tagja volt Franciaországban, a színészi mesterséghez azonban keveset értett. Hogy neve a XIX. században Shakespeare-rel kapcsolatban egyáltalán szóba került, az egy késői névrokonának, Delia Baconnak köszönhető, aki 1856-ban egy amerikai folyóiratban kifejtette magabiztos véleményét: nem egy egyszerű, vidékről felkerült kóbor színész írta a Shakespeare nevéhez fűződő darabokat, hanem a híres filozófus, a vitathatatlanul művelt Francis Bacon, aki szerényen megbújt egy álnév mögött. Az ötletet mások is felkapták, Appleton Morgan például 1881-ben egész könyvet írt róla „The Shakespearean Myth" címmel. A merész hipotézis később is többször felbukkant, az irodalomtudományban azonban a XX. század elején már senki nem vette komolyan.

A kérdés eldöntéséhez érdemes figyelembe venni a következőket:

1. Shakespeare valójában közel sem volt olyan művelt, mint Bacon. Éppen ezért követhetett el darabjaiban olyan hibákat, hogy Csehországot tengerparti országnak gondolta (Téli rege), Viennáról (vagyis Bécsről) pedig azt hitte, hogy itáliai városállam (Szeget szeggel). Ilyesmi Baconről egyszerűen elképzelhetetlen.

2. Semmi rendkívüli nincs abban, hogy az eredeti kéziratok nem maradtak fenn. Egyrészt tudjuk, hogy a Globe színház mindenestül leégett. Másrészt az idő tájt a szerzői jog fogalma szinte ismeretlen volt, főleg a színművekkel kapcsolatban. A szerző a színház megrendelésére dolgozott, vagy pedig eladta kész darabját egy-egy társulatnak. A színészek kézzel másolgatták ki belőle a maguk szerepét, teljes kéziratok csak a súgó számára készültek. Az óvatosságnak megvolt az oka: a teljes példányokat gondosan dugdosni kellett a konkurens társulatok elől, nehogy eltulajdonítsák.

3. Kalózkiadásban alkalmanként mégis megjelentek a legsikeresebb darabok. Ezek azonban rendszerint tele voltak hibával, mert vagy a nézők emlékezete alapján írták le a szöveget, vagy a színészek által ellopott súgói példányokból. Így Shakespeare 37 darabja közül életében 19 mű jelent meg kisalakú, ún. kvartó (negyedrét) formában, hibás szöveggel.

4. Shakespeare 1616-ban bekövetkezett hirtelen halálakor szörnyű rendetlenség maradt utána. Szerencsére két színésztársa, J. Heming és H. Condell felismerte, hogy pénzt lehet csinálni a hagyatékból, és nekiláttak a kéziratok rendezésének és javítgatásának. Hét év alatt készült el az összkiadás, benne a fellelhető 36 darabbal. (A Pericles csak később került a „kanonizált" darabok közé.) Az összkiadás kétszer akkora méretű, ún. fólió formátumban készült. (18 színművet egyébként kizárólag ebből a fóliókiadásból ismerünk.)

5. Külön története van a Morus Tamás (Sir Thomas More) című darabnak, amely meglehetősen zilált állapotban és itt-ott hézagosan maradt ránk. Ennek a darabnak Shakespeare csak egyik társszerzője volt (öten írták, valóságos nyomozómunkával a szerzőket is sikerült azonosítani). Nagy szerencse azonban, hogy a darab egyáltalán fennmaradt, mert a maga korában betiltották, csak az ellenőrzésre beadott példány került elő Erzsébet királyné színházi cenzorának, Edmund Tilneynek az irattárából. A kézirat külön érdekessége, hogy Shakespeare saját kezű javításait és betoldásait is tartalmazza, köztük a tragédia utólag hozzátett nagyjelenetét, amely kétséget kizárólag Shakespeare kezeírásával készült.

Maga a vita már jó száz éve elült, kérdés, hogy érdemes-e újra feleleveníteni. A kétkedőknek természetesen nem lehet megtiltani, hogy új számítógépes vizsgálatokkal próbálkozzanak: Bacon és Shakespeare szókincsét és stílusát össze lehet hasonlítani, bár szerintem előre megjósolható, hogy a Bacon műveiben használt szókincs meg sem közelíti Shakespeare műveinek szókincsét, ami kb. 15 ezer szóból áll.

Az érdeklődőknek külön figyelmébe ajánlom Szerb Antal háromkötetes világirodalom-történetének első kötetét (p. 335), ahol a szerző tőle szokatlanul éles jelzőkkel illeti ezt a hipotézist. Egyúttal megragadom az alkalmat, hogy köszönetet mondjak Mészöly Dezsőnek, aki megismertetett az ügy hátterével és Szerb Antal figyelemre méltó véleményével.

Vargha Dénes

Nem igazán a mi lapunk profiljába tartozik, hogy a világirodalom talán legnagyobb szellemi teljesítményének tekinthető művek szerzőjéről régóta tartó vita részleteiben elmélyedjünk. Az viszont egy kicsit a mi ügyünk is, hogy az összehasonlító nyelvi elemzés legújabb számítógépes módszerei hogyan adhatnak a kutatásnak új lendületet.

A probléma történeti bonyadalmai és nyelvészeti finomságai miatt a kutatásban elsősorban persze maguknak az angoloknak kell jeleskedniük. Véleményemet én sem saját forrásokra vagy önálló elemzésre alapoztam, hanem arra, hogy a korábban már felsorakoztatott érvek és ellenérvek mellett az utóbbi időkben előkerült dokumentumok és a legújabb összehasonlító vizsgálódások megítélésem szerint egyre jobban alátámasztják azt a hipotézist, hogy William Shakespeare neve alatt, egy valós kortárs személy mögött meghúzódva e műveket mégiscsak Francis Bacon írta.

Vargha Dénes a fentiekben vázolta, miért nem tartják igazoltnak a Bacon-teóriát. Ez a vita sem tekinthető azonban lezártnak. Akit érdekel, hogy miért nem, és milyen érvek és dokumentumok állnak szemben a jelenleg még hivatalos Shakespeare = Shakespeare állásponttal, az látogasson el a http//www.sirbacon.org webhelyre, melynek néhány oldalát CD-mellékletünkre is feltettük, és onnan a megfelelő linkekre rákattintva a weben lehet folytatni a témával való ismerkedést.

Faklen Pál

Feladványok és adatbázisok

Amikor a számítógép mondja ki a végső szót

A ChessBase néven ma már világszerte ismert, szinte minden sakkozónak a versenyekre történő felkészülését segítő program ötletéről 1979-ben hallottam először. Ez az adatbázis lényegében a jelentős versenyeken és párosmérkőzéseken lezajlott játszmák folyamatosan bővített gyűjteménye. Frederic Friedel fiatal német tudományos újságíró több mint 20 évvel ezelőtt mondta el a sakkadatbázis ötletét a Spiegel egyik vezető munkatársának, majd később közös céget alapított Matthias Wüllenweber számítástechnikussal, aki összehozott egy kiváló programozó csapatot.

Friedelnek a sakkadatbázisra vonatkozó 1979-es elképzelése eleve hatalmas léptékű volt. Nem sokkal a nyilvános bemutatás után már mintegy 300 ezer partit gyűjtöttek össze, napjainkig pedig mintegy 2 milliót.

A ChessBase legfontosabb sajátossága, hogy a játszmák többféle szempont szerint lehívhatók, így a versenyzők tanulmányozhatják azokat például a megnyitás jellege, a sakkozók személye, a versenyek helyszíne stb. szerint. Verseny előtt a leendő ellenfelek korábbi játszmái ezzel a programmal kényelmesen elemeztethetők.

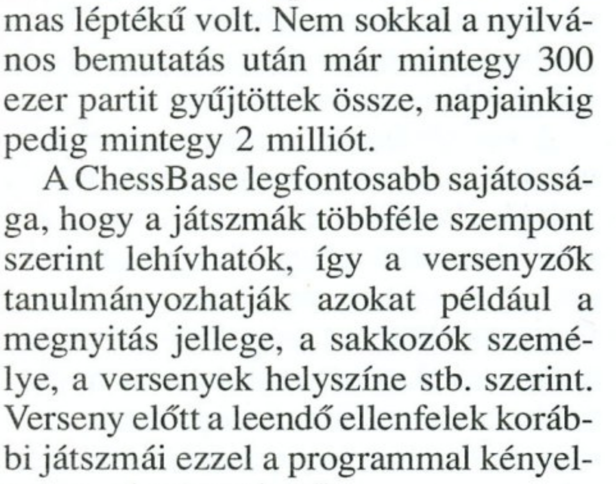
A ChessBase után hasonló elven működő más játszmaadatbankok is megjelentek (NicBase, Chess Asssistant). Az Új Alaplapban már bemutatott (és demóváltozatban a CD-n most is közreadott) Superpro adatbázist Lovass László teljesen új alapokon, a „sakkfa" elve szerint építette fel. A nyitólépésre a potenciális ellenlépések, majd az ezekre adható válaszok, és a további lehetséges válaszok exponenciálisan növekvő számban alkotnak új hadállásokat. Ezeknek persze csak kis hányada reális, és még kevesebb a napjainkig lezajlott partikban konrétan létrejött hadállás. A Superpro program jelenleg mintegy 4 millió virtuális játszmát tartalmaz, amelyek elemzését a ténylegesen lezajlott partikkal össze lehet hasonlítni.

Adatbázist alkothatnak a meghatározott anyagerősséget vagy bábszámot tartalmazó hadállások is, elsősorban persze a kevés figurára redukált végjátékok. Egy-egy ilyen adatbázis is lehet persze nagyméretű, azok tartalma mégis véges. Példaként említem a királyok mellett egy–két–három gyalogot felvonultató végjátékokat. Miként ma már gyakorlatilag szinte minden sakkhadállás esetében, megfelelő programmal ezeknél is kimutatható, hogy a jobb helyzetben lévő megnyerheti-e a játszmát, vagy döntetlen a parti. Ezek a hadállások bizonyos szabályszerűségeknek engedelmeskednek, amelyekre a program megtanít bennünket. Részletes ismertetésük nemcsak ezen cikk, hanem egy könyv terjedelmét is meghaladná. A következőkben először néhány érdekes illusztrációt említek meg a gyakorlati szempontból is fontosnak bizonyult négy- és ötbábos hadállások közül, vagyis amikor a két király mellett legfeljebb három további báb van, és azok elhelyezkedése egy játszma alapállásából levezethető.

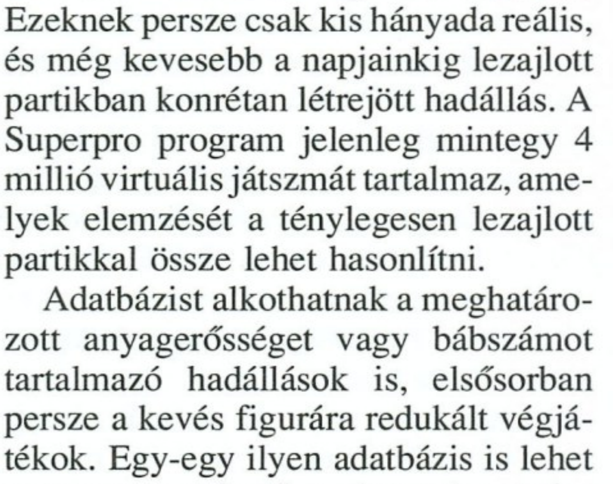
Számítógépes programmal minden szabályszerűen lehetséges négy- és ötbábos hadállást mindkét fél oldaláról tökéletesen kielemeztek. A négybábos hadállások közül komoly feladat csak az, amelyben vezér küzd bástya ellen. A gép kimutatta, hogy az ilyen állás megnyerhető, és bármely helyzetben megmutatja ennek legrövidebb útját is. Az ötbábosok sorában az alábbi konfiguráció végigelemzése a sakktörténet egyik különösen érdekes fejezete, a számítógépes sakknak pedig igazi „sikerélménye".

A történet azzal kezdődik, hogy a múlt század derekán a kiváló német sakkszerzőpáros (Kohtz és Kockelkorn) feladta a leckét: amikor egy végjátékban az 1. ábrán látható hadállás alakul ki, tudja-e tartani sötét a döntetlent egy huszárral két világos futó ellenében. A szerzők kimutatták, hogy igen, mégpedig úgy, hogy a király a b2 (vagy b7, g2, g7) mezőt foglalja el, a huszár pedig a király körzetében manőverez, amelyet ez esetben nem lehet két futóval sarokba szorítani és megmattolni.

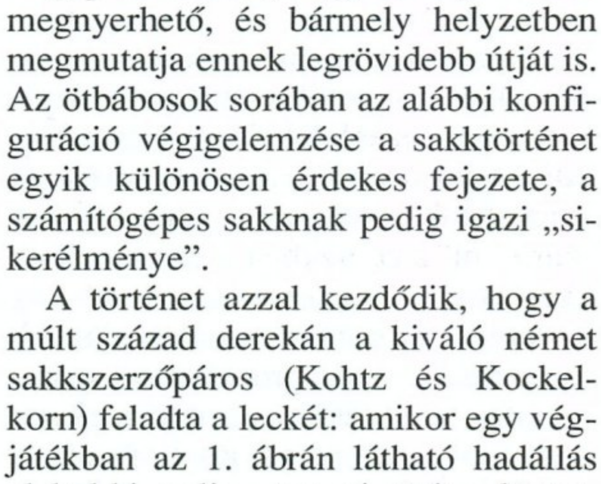
Ez a két futó kontra huszár konfiguráció évekig izgatta a végjátékszakértők



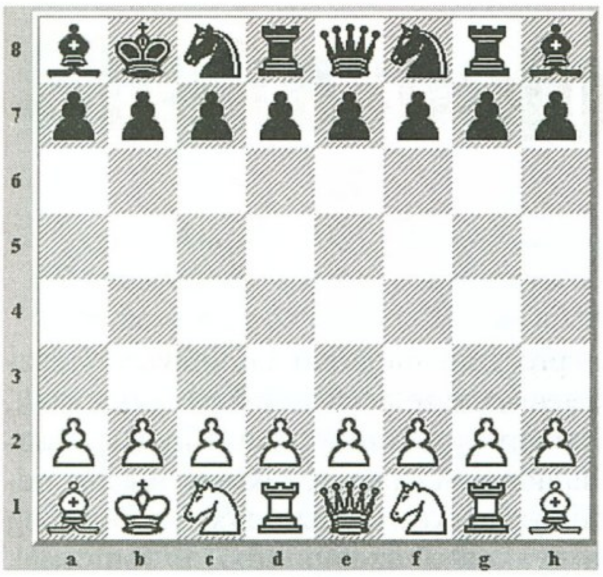
1. ábra



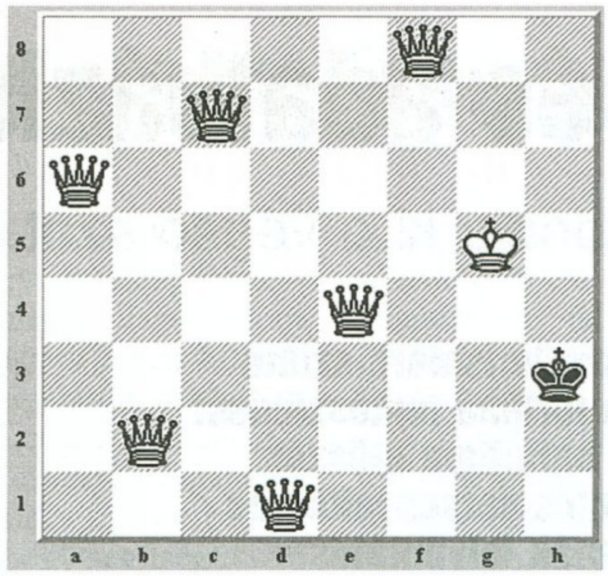
2. ábra



3. ábra



4. ábra



5. ábra

50	11	24	63	14	37	26	35
23	62	51	12	25	34	15	38
10	49	**64**	21	40	13	36	27
61	22	9	52	33	28	39	16
48	7	60	**1**	20	41	54	29
59	4	45	8	53	32	17	42
6	47	2	57	44	19	30	55
3	58	5	46	31	56	43	18

6. ábra

és a programozók fantáziáját. 1980-ban azután Ken Thompson amerikai professzor, a számítógépes sakk egyik nagy alakja (alkotása, a Belle évekig volt a számítógépes sakk világbajnoka) programjával teljes elemzés kíséretében kimutatta, hogy a német páros következtetése téves, a sötét király ebben az esetben is kimozdítható helyzetéből, és a mattot a futókkal ki lehet kényszeríteni.

A gyakorlati partik gyakran torkollnak bástyavégjátékba, és ezek a sakkjáték legnehezebb feladatai közé tartoznak. Itt most szintén csak az ötbábosok csoportjáról ejtek szót, vagyis olyan állásokról, amelyekben a két király és a két bástya mellett maradt egy gyalog is a táblán. Az egykori Szovjetunió több végjátékszakértője próbálkozott olyan rendszer felállításával, amely segíti az ilyen helyzetek megoldását. A végső ítéletet azonban csak a számítógép mondta ki, felsorolva rengeteg nyerő hadállást a győzelemhez vezető úttal, és rengeteg másikat, amelyben a gyalogelőny nem érvényesíthető. Gyakori eset, hogy egyetlen báb áthelyezése ellenkező eredményre vezet. Ezt illusztráló példát mutatok be a 2. ábrán.

Ebben a helyzetben világos 1. Kg6 Bb6+? 2. f6 Bb8 3. Bh7 Kg8 4. f7+ Kf8 5. Bh8+ Ke7 6. Bxb8 útján nyer. A sötét bástya alulról történő ellentámadása azonban menti a helyzetet: 1. — Bb6+! 2. f6 Bb1! és az örökös sakk elől nincs menekvés. Érdekes, hogy ha a világos király az alaphelyzetben egy mezővel előrébb áll, akkor 1. Kg6! nyer, mert 1. — Bb1-re 2. Ba8#. A bástyának a 8. soron kell maradnia, és például 1. — Bd8-ra 2. Bh7 a fent kimutatott módon nyer.

A eddigiekben bemutatott két konfiguráción kívül csak két további „problémás" akad, a többiben a parti végkimenete nyilvánvaló. Ha például világos vezérrel és bástyával vagy két bástyával rendelkezik, bármi is az egyetlen sötét báb, világos nagyon csekély kivétellel biztos nyerő. Viszont igen ritka a nyerés, ha közepes erősségű két bábbal bástya vagy vezér áll szemben. Ilyenkor gyakran ismét csak a számítógép segítségével lehet megállapítani a mindkét fél számára legjobb lépéseket (néha több egyenértékűt is).

Bástya és futó a másik egyetlen bástyája ellenében minden helyzetben nyer, nem is túl nehéz a bástyát a futó és a király segítségével az egyik szélső sorra vagy oszlopra leszorítani, és a mattot vagy az ellenfél bástyájának lenyerését kikényszeríteni. Legnehezebben az a különleges konfiguráció ítélhető meg, amelyben két huszárral szemben az ellenfélnek egyetlen gyalogja marad a táblán. Ilyen a 3. ábrán látható hadállás.

Kezdő sakkozók is tudják, hogy pusztán két huszárral nem lehet mattot adni, mert a sarokba szorított királyt előtte patthelyzetbe kellene hozni, vagyis a parti döntetlen. Viszont a gyakorlati játszmákban előfordul, hogy az ellenfélnek marad egy szabadon mozgó gyalogja a táblán, és annak szabad lépése lehetővé teszi, hogy a két huszár a sötét királyt patthelyzetbe hozza, hogy utána mattot adjon. Ez történik itt is. A nyerés módja egyszerű: 1. H4f3+ Kh1 2. He4 (ebben a pillanatban patt lenne, ha nem lenne egy sötét gyalog is a táblán) 2. — d2 5. Hf2#. Hasonló a helyzet, ha a sötét gyalog más olyan mezőn áll, ahonnan megtehet egy szabad lépést. Ugyanígy kell eljárni, ha a d4 világos huszár e1-en vagy g5-ön áll, sőt akkor 1. Hf3 mellett 1. Hh3+ is célhoz vezet. Megintcsak a számítógép mutatta ki, hogy melyek az ilyen egyedinek mondható helyzetek, és mivel azok száma véges, fel is sorolhatók. Parti közben ehhez hasonló állás elérésére kell törekedni, ami nem könnyű, de egy jó sakkozó megtalálja a módját.

Az eddigiektől eltérő típusú adatbázist alkotnak a újabban divatossá váló „fischerandom" partik alapállásai. A kifejezést a Fischer és a random (véletlen) összevonásából alkották, ugyanis Bobby Fischer ötlete volt, hogy a sakkot változatosabbá lehetne tenni, ha a partikat nem a megszokott alaphelyzetből indítanák, és csak a gyalogok állnának ugyanúgy a második, illetve hetedik soron, mögöttük a tiszteket véletlenszerűen, esetleg sorsolás alapján helyeznék el, persze bizonyos megkötések mellett. Ezt szemlélteti az 4. ábrán bemutatott indulóállás.

A két fél bábjainak megegyező felállítása nem okvetlenül követelmény. Tulajdonképpen minden olyan esetben sakkadatbázist hozunk létre, amikor a sakkjáték szabályainak megfelelő partikat, hadállásokat vagy lépéssorozatokat gyűjtünk egybe meghatározott feltételek szerint. Hosszú ideje ismert például a feladat, hogy helyezzünk el a táblán nyolc (illetve a két király mellett hat) egyszínű vezért úgy, hogy azok ne essenek egymás ütőkörébe (5. ábra). J. Gik orosz matematikus és sakkozó könyvet írt az efféle sakkállásokról, néhány példát ebből merítettem. Ezeknek a gyakorlati sakkhoz csak annyi közük van, hogy a bábok a sakkszabályok szerint mozognak, a megoldás azonban nem „sakkszerű", hanem matematikai gondolkodással történik.

Hasonló, de nehezebb lecke: egy huszárral bármely mezőről elindulva kell az egész sakktáblát bejárni. E feladatok korrekt megoldásával lényegében szintén sakkadatbázisokat hozunk létre. Ennek a példának egyik megoldása a 6. ábrán látható.

Hasonló feladatok, melyekben már a matematika játssza a főszerepet, a számítógépes sakknak köszönhetően igen sokféle változatban megalkothatók, és magához a klasszikus sakkjátékhoz hasonlóan szinte kimeríthetetlen lehetőségeket tárnak fel.

Lindner László
linchess@elender.hu

Szelídítsünk pingvint (VI.)

Hasznos segédprogramok és apró ötletek

Sorozatunknak ebben a részében néhány olyan segédprogramot mutatunk be, amelyek az eddigi tárgyalásból kimaradtak, éppen azért, mert nem kapcsolódtak szorosan egyik cikk tematikájához sem. Emellett összegyűjtöttünk egy csokorra való apró, de hasznos ötletet is.

Winchesterhangolásra: hdparm

Az első bemutatandó utility a hdparm. Ez egy olyan segédprogram, amellyel az IDE vagy EIDE felületű merevlemezek működési paramétereit lehet beállítani. Ennek a legfontosabb célja az lehet, hogy minél gyorsabb működésre bírjuk az eszközt, amivel persze a Linux-alkalmazások sebességét is érezhető mértékben növelhetjük. A program része a Red Hat disztribúciónak, az rpm csomag az rpm-ivh hdparm*rpm paranccsal telepíthető.

Mielőtt a hangoláshoz hozzákezdenénk, érdemes egy sebességtesztet elvégezni a programmal. A -t kapcsoló a pufferolt olvasás, míg a -T kapcsoló a buffer cache olvasási sebességét méri. A parancssorban meg kell adni persze a meghajtó eszközt is, a hdparm -t /dev/hda például az első IDE vezérlő elsődleges eszközének sebességét méri. Írjuk fel ezeket az értékeket, hiszen csak olyan paramétereket érdemes majd megváltoztatnunk, amelyekkel javíthatók a tesztek által mért értékek.

Nagyon sok paraméter beállítására van lehetőség, némelyiknek a használata kockázatos is lehet, ezért csak azokat célszerű alkalmazni, amelyeket a man dokumentáció is ajánl. Ezek az alábbiak:

```
c : 32 bites i/o-támogatás
A : előolvasás engedélyezése
m : multiszektor i/o (érdemes 16-ra állítani)
d : DMA-használat engedélyezése
a : előolvasó pufferméret (8 vagy 4 az ajánlott,
lehet, hogy a 4 gyorsabb működést eredményez, ha sok
apró fájlt kell olvasni)
W : írási cache
```

Ezeken kívül érdemes még megemlíteni a következő opciókat is:

```
E : a CD-ROM sebességbeállítása
r : a merevlemez csak olvashatóvá tétele
S : a lemez automatikus leállítási idejének megadása
z : az automatikus lepörgés tiltása
```

Egy alkalmas parancssor például az alábbi lehet:

```
hdparm -c 1 -A -m 16 -a 4 -d 1 /dev/hda
```

Ez a beállítás egy 2 Gbájtos Seagate winchesteren másfél-kétszeres sebességnövekedést eredményezhet, ami már észrevehető például az X Window elindulásakor is. Mivel a paramétereket minden rendszertöltéskor be kell állítani, ezért a hdparm indítását el kell helyezni valamelyik rendszerindító állományban. Érdemes már a betöltés elején, minél korábban elvégezni az átállítást, mert így a betöltési folyamat hátralévő része is gyorsabb lesz.

Az egyik megoldás az rc.sysinit rendszerindító állomány módosítása. Ez a Red Hat 7.0-ban az /etc/rc.d könyvtárban található (a 7.0 némileg változtatott az indító szkriptek elhelyezésén). Legjobb, ha az indító parancssort közvetlenül a PATH beállítása után helyezzük el. Ha több merevlemezünk is van, akkor több hdparm hívásra lesz szükség. Ha azt szeretnénk, hogy a hdparm ne írogasson a képernyőre, akkor helyezzünk el minden kapcsoló elé egy -q kapcsolót, vagy irányítsuk át a kimenetet a /dev/null eszközre.

A videomód beállítása: xvidtune

Ez a segédprogram a monitor és a grafikus üzemmód paramétereinek összehangolására alkalmas. Amikor telepítjük és konfiguráljuk az XFree86-ot, akkor az egyes videomódokhoz tartozó paramétereket a Linux egy paraméterfájlból veszi. Ezek az alapértékek nem biztos, hogy megfelelnek az általunk használt monitor és videokártya párosnak, emiatt a grafikus képernyő a monitoron eltolódhat vagy torzulhat. A kép pontosabb beállítására alkalmas az xvidtune, amely az XFree86 videomód kiterjesztésén keresztül teszi lehetővé a hangolást.

A program X Window alatt fut, legegyszerűbben az Alt+F2-re megjelenő miniparancssorból indítható, az xvidtune begépelésével. A program elindulásakor figyelmeztet, hogy bizonyos szélsőséges értékek használata egyes monitorokat tönkre is tehet (régebbi típusú monitorok esetén érdemes megfontolni ezt a figyelmeztetést). A kép pontos beállítása a következőképpen történik: kapcsoljuk be az Auto üzemmódot, ekkor ugyanis az értékek megváltoztatásával azonnal módosul a kép is, így a korrekció egyszerűbb és gyorsabb. A kép méretét és pozícióját a megfelelő nyomógombokkal (Left, Right, Wider, ...) állíthatjuk be.

Bármilyen tévesztés esetén az R billentyűvel hozhatjuk vissza az előző stabil beállítást. Ha a kép mérete és helyzete is megfelelő, a Show nyomógomb lenyomására a program a standard kimenetre írja ki a beállításnak megfelelő értékeket. Ezeket az értékeket kell majd beírnunk az XFree86 konfigurációs állományába, az /etc/X11/XF86Config fájlba. Hogy a munkát megkönnyítse, az xvidtune pontosan ugyanolyan formában adja meg a paramétereket, amilyenben az egyes Modeline sorok azt megkívánják. Már csak azt kell kiderítenünk, hogy a konfigurációs fájlban található sok Modeline sor közül melyiket kell javítanunk. Ehhez azt kell tudni, hogy a Linux az adott felbontáshoz az egyes Modeline sorok közül a monitorunk képességei szerinti legjobb beállítást választja, ezért ebben kell majd javítanunk. Nézzük meg a következő példát: 800x600-as felbontást használunk, és a monitorunk paraméterei: HorizSync 31.5–48.5 kHz, VertRefresh 50– 90 Hz.

Emeljük ki az XF86Config fájlban a 800x600-as felbontáshoz tartozó paramétersorokat:

800x600 @ 56 Hz, 35.15 kHz hsync Modeline "800x600" 36 800 824 896 1024 600 601 603 625

800x600 @ 60 Hz, 37.8 kHz hsync Modeline "800x600" 40 800 840 968 1056 600 601 605 628 +hsync +vsync

800x600 @ 72 Hz, 48.0 kHz hsync Modeline "800x600" 50 800 852 972 1060 600 601 607 622 +hsync +vsync

800x600 @ 85 Hz, 55.84 kHz hsync Modeline "800x600" 60.75 800 864 928 1088 600 616 621 657 -HSync -VSync

800x600 @ 100 Hz, 64.02 kHz hsync Modeline "800x600" 69.65 800 864 928 1088 600 604 610 640 -HSync -VSync

Látható, hogy a harmadik paramétersor a monitorunk képességeinek megfelelő legjobb beállítás (a negyedik sorban az 55.84 kHz-es vízszintes frekvencia már meghaladja a monitorra megadott 48.5 kHz felső határt, ezért ez már nem használható). Azaz a harmadik sorban kell az értékeket az xvidtune által megadottakra javítanunk (a fenti listában már a korrigált paraméterek láthatók).

A konfigurációs fájl „kézzel" történő javítása talán kissé ódivatúnak és körülményesnek tűnhet, de ennyi kényelmetlenséget biztosan megér egy tökéletesen középre pozicionált és méreteiben is optimális grafikus képernyő.

A lemeztelítettség monitora: kdumon

Egy lelkiismeretes rendszergazda egyik fontos feladata a fájlrendszer szabad területével való gazdálkodás. Ettől függ, hogy mennyi hely marad az egyes felhasználóknak, milyen alkalmazásokat lehet még telepíteni, esetleg melyektől kell mindenképpen megválni a szűkös hely miatt. Érdemes néhány naponta ellenőrizni a rendelkezésre álló szabad területet, hiszen a Linux működés közben gyarapítja a fájlrendszert a naplóval és az átmeneti állományokkal (erről később még szót ejtünk), ami esetleg feleslegesen csökkenti a felhasználható lemezterületet.

A lemez telítettségének gyors és kényelmes áttekintéséhez kiváló segédeszköz a kdumon. A program egy fa struktúrában ábrázolja az egyes könyvtárakat a bennük levő fájlok által lefoglalt összes mérettel együtt. Ott bármelyik könyvtárat kibonthatjuk, becsukhatjuk, így pillanatok alatt kideríthető, hogy melyek azok a könyvtárak, amelyek a legterjedelmesebbek, és melyekre férne rá egy alaposabb tisztogatás.

Az alkalmazás elérhető a KDE WebCity webhelyen, a http://devel-home.kde.org/~kdumon címen. Célszerű az rpm csomagot letölteni, mert ez minden gond nélkül, egyszerűen installálható a már ismert módon. Fontos tudni, hogy a program a du utility segítségével gyűjti össze a megjelenítendő adatokat, így ennek elérhetőnek kell lennie a PATH alapján.

Még egy fontos megjegyzés kívánkozik ide. Ha már megállapítottuk, hogy mely könyvtárak szorulnak rá egy kis fogyókúrára, ne kezdjünk meggondolatlanul a tisztogatásba. Egészen ártalmatlannak és feleslegesnek látszó fájlok eltávolítása is járhat kellemetlen következményekkel: nem indul el az X Window, nincsenek ékezetek, váratlan hibaüzenetek jelennek meg stb. Az ajánlható módszer az, hogy a kiválasztott fájlokat tömörítsük eredeti helyükön és egyenként. Erre alkalmas például a Midnight Commander felhasználói menüje, ahol a bz2 tömörítést választva nagyon jó tömörítési arányt érhetünk el. A fájlok nevét a tömörítő a .bz2 taggal egészíti ki, és ez az állomány szükség esetén pillanatok alatt visszaalakítható.

Válogatás és tömörítés után használjuk üzemszerűen a rendszert, néhány heti hibamentes használat esetén a tömörített fájlok is nyugodtan törölhetők. Persze egy verzióváltás, egy csomag újratelepítése vagy frissítése ismét visszateheti a feleslegesnek ítélt és már eltávolított állományokat, ezért ilyenkor a fenti eljárást érdemes megismételni. Erre persze főleg azoknak van szükségük, akik winchesterükön helyhiánnyal küzdenek, vagy szeretik helytakarékosan karbantartani rendszerüket.

Lapozás a konzolon

Gyakran okoz kellemetlenséget, hogy egy konzolra író programot elindítva a várt rövid, egyképernyős kiírás helyett hosszabbat kapunk, így az lefut a képernyőről. A tettes lehet az ls, a tar, de a többi konzolalkalmazás is. Ilyenkor az egyik megoldás természetesen az, hogy újra beírjuk a parancsot lapozó paraméterrel (ilyen például a more vagy a less, például: ls -la | more). Egy másik lehetőség a Shift+PageUp billentyűvel visszalapozni a konzolon, és megnézni a képernyőnek azt a részét, amely egyébként már nem lenne látható (a Shift+PageDown az előrelapozó). Ez sem alkalmas azonban tetszőlegesen hosszú kimenet visszalapozására, mert a konzol a kimenet tárolására korlátozott nagyságú memóriaterületet használ. Egy átlagos directorylista esetében 10-12 lap fér el ezen a területen. Érdemes még megjegyeznünk, hogy konzolváltás után ez a memóriaterület kiürül, így visszalapozni sem tudunk akkor, ha visszaváltunk az eredeti konzolra.

Kevesebb konzol

A Linux alapértelmezésben 6 szövegalapú konzolt használ. Mivel igen ritka esetben szükséges ennyi konzol egyidejű használata, ha egy kis memóriát szeretnénk felszabadítani egyéb célokra, érdemes a konzolok számát csökkenteni. Egy átlagos felhasználó számára két-három karakteres és egy grafikus konzol tökéletesen elegendő. De legyünk nagyvonalúak, és hagyjunk meg 4+1 konzolt.

A Linux az indítás során az /etc/inittab állományban indítja el az egyes konzolok kezelőprogramjait, így ebben az állományban kell javítanunk. (Az eredeti fájlt persze érdemes elmenteni.) Keressük meg azt a részt, ahol az indítások történnek, és tegyünk megjegyzésjelet az 5. és 6. konzol sora elé:

```
#5:2345:respawn:/sbin/mingetty tty5
#6:2345:respawn:/sbin/mingetty tty6
```

Újraindítás után a ps-ax paranccsal meggyőződhetünk arról, hogy már csak négy konzol aktív. Nagyon fontos, hogy ezután már nem az Alt+F7 billentyűvel tudunk a grafikus konzolra átváltani, hanem az Alt+F5-tel.

A core fájlok készítésének tiltása

Abban az esetben, ha egy program bármilyen hiba miatt rendellenesen áll le, a Linux kernel alapértelmezésben kezdeményezi a program által használt memória tartalmának fájlba írását. Ez az ún. core dump, a fájl pedig az aktuális könyvtárba kerül core néven. Ez a fájl igen nagy is lehet, és ha sok készül belőle, akkor feleslegesen foglalja a helyet a winchesteren. Ha a Linux rendszer nem stabil, mert kísérleti állapotban lévő kernelt vagy meghajtót használunk, illetve valamelyik általunk telepített alkalmazás sokszor áll le rendellenesen, a core fájlok letakarítása rendszeres teendővé válhat.

Ha nem szándékszunk ezeket a fájlokat hibakeresésre használni (ami nyilván nem egyszerű dolog), és nem akarunk ezek eltávolításával bajlódni, legegyszerűbb, ha jelezzük a kernelnek, hogy nem kérjük a core dumpot. Erre az ulimit programot használhatjuk, amely a bash shell egyik beépített programja, és bizonyos korlátozások beállítására való. A core fájlokkal kapcsolatban csak annyit kell tudnunk, hogy a -c 0 kapcsolóval ezek automatikus létrehozása letiltható. Nincs is más teendőnk, mint ezt a hívást egy erre alkalmas helyre elhelyezni. Egy megoldás, ha az /etc/bashrc állományba

valahová beírjuk az ulimit -c 0 parancsot. Ez a továbbiakban megkímél bennünket a core fájlok vadászatától.

Új vagy megváltozott állományok figyelése

A Linux a mindennapos használat során a már említett core fájlokon kívül más állományokat is készít automatikusan. Naplóállományok, man oldalak cache állományai, böngésző és fájlkezelő cache, átmeneti állományok. A kernelfordítás során is sok új állomány készül, és egyes alkalmazások telepítésekor is olyan új fájlok kerülhetnek a winchesterünkre, amelyek esetleg elkerülik a figyelmünket. Mindennek következményeként azt tapasztaljuk, hogy a naponta lefuttatott df parancs egyre kevesebb szabad helyet mutat. Érdemes tehát az új állományok keletkezését valahogy nyomon követni. Ehhez a find parancsot hívhatjuk segítségül, és hogy ne kelljen a parancssori kapcsolókat fejben tartani, készítsünk egy szkriptet, amelyben megadható, hogy hány napra visszamenően listázza az új vagy megváltozott fájlokat. Legyen ez a szkript például a list_new_files, és tegyük azt olyan könyvtárba, amely szerepel a PATH beállításban. A szkript a következőképpen néz ki:

```
#
#!/bin/bash
#
#  list_new_files : List New Files
#
#  Usage : list_new_files ays
#
find / -mount -mtime -$1 -printf "%p\t%s\t%t\n"
```

A find a gyökérkönyvtártól kezdve, csak a Linux fájlrendszerre korlátozottan minden alkönyvtárban megkeresi azokat az állományokat, amelyek az utóbbi napokban keletkeztek vagy változtak, kiírja ezek nevét, hosszát, a keletkezés vagy a változás idejét tabulátorokkal elválasztva.

A napok számát paraméterként adhatjuk meg az eljárásnak. Az utóbbi két napban keletkezett vagy módosult állományok a list_new_files 2 paranccsal listázhatók ki.

Programindítás egyetlen billentyűvel

A DOS operációs rendszerben a command.com helyett használt alternatív parancsértelmezőkben (ilyen az NDos, a 4Dos) egyetlen billentyűhöz utasításláncot hozzárendelve gyorsindító billentyűket lehetett létrehozni az alias helyettesítő utasítás segítségével. Bár ez tényleg csak egy kényelmi szolgáltatás, de aki megszokta, az Linux alatt is keresni fogja. Hiszen nagyon sok olyan parancs van, melyet naponta sokszor kell beírni és Enterrel lezárni. Ilyen például a dir, ls, mc, x, logout. De érdemes néhány ritkábban használt nevet (halt, reboot) is gyorsbillentyűhöz kötni.

Akik ismerik a Linuxot, vagy éppen sorozatunk által kerültek közelebb hozzá, valószínűleg sejtik, hogy van rá megoldás. A bash shell lehetőséget nyújt ilyen gyorsbillentyűk definiálására is. Akik nem ezt a shellt használják, azoknak nem csupán ezért, hanem a bash által kínált sok más szolgáltatás miatt is érdemes megfontolni az áttérést erre a parancsértelmezőre.

Nézzük a konkrét megvalósítást. A Linux bash shellje az inputrc állományt használja az egyes billentyűk funkciójának megváltoztatására. Ennek helye az /etc könyvtár. Mégse ide tegyük a gyorsbillentyűk definícióit, erre alkalmasabb a bejelentkezési vagy home könyvtár. Így minden felhasználó számára külön kiosztást alakíthatunk ki. Az egyetlen eltérés, hogy itt az állományt .inputrc-nek hívják. Logikus és hasznos konvenció, hogy az /etc könyvtár állományainak megfelelő állományok a home könyvtárban is elhelyezhetők, és ezek neve ponttal kezdődik. Ezek az állományok mindig felülbírálják az /etc-ben lévő párjuk beállításait.

Készítsünk tehát home könyvtárunkban egy .inputrc állományt, mondjuk az alábbi tartalommal (ha már van ilyen állomány, akkor természetesen a meglévőt egészítsük ki):

```
#
# Gyorsbillentyűk -- Key bindings
# F5 az mc indításához -- F5 for starting mc
"\e[[E": "mc \C-m"
# F6 az X Window indításhoz -- F6 for starting x
"\e[17~": "x \C-m"
# F10 a kijelentkezéshez -- F10 for logout
"\e[21~": "logout \C-m"
# F11 : újraindítás -- F11 for reboot
"\e[23~": "reboot \C-m"
# F12 : leállítás -- F12 for halt
"\e[24~": "halt \C-m"
```

A \C-m szekvencia azt jelenti, hogy Enter sem kell a parancssor lezárásához, az adott parancs a billentyű leütésekor elindul. Ha tovább szeretnénk a fájlt bővíteni, természetesen tudnunk kell az egyes billentyűk kódjait. Ezt a következő trükkel tudhatjuk meg. Ha például az F4 billentyűhöz szeretnénk parancsot kötni, nyomjuk le az Esc, majd az F4 billentyűt. A parancssorban megjelenik az [[D karaktersor, ezt kell beírni a \e után a kötést definiáló sorba.

Természetesen ezek a gyorsbillentyűk csak a bash shellben lesznek érvényesek, tehát ha elindítjuk a Midnight Commandert, az F6 a ott a megszokott módon fog működni.

Egyedi prompt a Midnight Commanderhez

Ha már a Midnight Commanderről szó esett, érdemes egy apró trükköt ezzel kapcsolatban is megismernünk. Én magam a Commandert igen sokat használom a könyvtárak felderítéséhez, átrendezéséhez és szövegszerkesztéshez. Ha programfuttatás során szeretném a program kimenetét látni, Ctrl+O-val eltüntetem a Commander képernyőjét. Igen gyakran előfordul, hogy nem állítom azonnal vissza egy újabb Ctrl+O-val a eredeti képernyőt, emiatt hajlamos vagyok azt is elfelejteni, hogy már fut egy mc a konzolon, és esetleg elindítok egy újabb példányt belőle. Ez tényleg apróságnak tűnhet, de egy nap többször elkövetve ezt a hibát, meglehetősen bosszantó tud lenni. Jó lenne, ha a konzolról akkor is leolvasható lenne az mc jelenléte, ha képernyője nem látszik.

Erre legjobb eszköz a prompt, a shell készenléti jele. Jelezhetjük a prompt szövegében is az mc jelenlétét, de talán még szembetűnőbb, ha a prompt színét változtatjuk meg. Ehhez a bash shell SHLVL azaz shell level (parancsértelmező szint) változóját használjuk. Ez az elsődleges értelmezőben lekérdezve 1-et ad vissza, minden további shellindítás pedig eggyel növeli az értékét.

Nézzük ezután a megoldást. Home könyvtárunkban készítsünk egy .bashrc fájlt (vagy egészítsük ki a már meglevőt) az alábbi tartalommal (csak emlékeztetőül: a .bashrc-nek is van /etc-beli megfelelője, a bashrc):

```
if [ $SHLVL -ne 1 ]; then
   export PS1="\[\033[1;33m\]\u \w\[\033[0m\]"
fi
```

Ez a kódrészlet minden, nem elsődleges shellindításkor a készenléti jelet sárga színnel írja ki. Ha mondjuk az egyébként használt prompt szín a zöld, akkor ez elegendően figyelmeztető értékű lehet, így ezt a promptot látva nem indítunk újabb Commandert, hanem Ctrl+O-val visszakapcsoljuk a már futó mc képernyőjét.

Szűcs János
szucsj@josa.szabinet.hu

A szálakról

Java tanfolyam haladóknak — VII. rész

A mostani részben a többszálú feladatok végrehajtásról, a Java ütemezési modellről és az éhező kínai filozófusokról fogok beszélni.

Többszálú programvégrehajtás

A mai operációs rendszerek képesek arra, hogy egyidejűleg több feladatot hajtsanak végre. Természetessé vált számunkra, hogy egy szövegszerkesztőben új dokumentumot szerkesztünk, vele párhuzamosan egy másik programból valamit kinyomtatunk, közben a számítógépen zenét is hallgatunk, egy ablakban pedig figyeljük, hogy elkezdődött-e már a tévében az a műsor, amelyet okvetlenül meg akartunk nézni. Ezek mind külön programok, azonban nem szálak, hanem processzek, vagyis az operációs rendszer alapvető végrehajtási egységei. Ha számítógépünknek egyetlen processzora van, akkor a kernel időszeletekre (time slices) osztja fel a rendelkezésre álló processzoridőt, és azt igazságosan (vagy igazságtalanul) szétosztja. Mivel az időszeletek nagyon rövidek, a felhasználó számára úgy tűnik, mintha a programok párhuzamosan futnának, holott valójában egymás utáni, szekvenciális végrehajtás történik.

Többszálas végrehajtásról (multithreaded execution) akkor beszélhetünk, ha a folyamat egy processzen belül történik. A szálaknak is van kezdete, végrehajtási része és vége, akár a processzeknek. Egy szál viszont nem tekinthető önálló programnak, hiszen önmagában nem indítható el. A szálak párhuzamosan futó szekvenciális feladatok, amelyek egy programon, nem pedig egy operációs rendszeren belül hajtódnak végre.

Java szálak létrehozása

Minimálisan egy osztályt és egy interfészt kell használnunk ahhoz, hogy működő szálat kapjunk. Elsőként meg kell írnunk egy olyan osztályt, amely megvalósítja a java.lang.Runnable interfészt:

```
class RunnableClass implements Runnable {
   public void run() {
   }
}
```

Ahogy látjuk, ez az interfész egyetlen metódus megvalósítását teszi kötelezővé, a run( ) metódusét. Ebbe az eljárásba kell majd beírnunk azt a kódrészletet, amelyet végre akarunk hajtatni a szál elindítása után. Programunkban valahol még létre kell hozni a java.lang.Thread osztály egy példányát, és konstruktorának paramétereként át kell adnunk azt az osztályt, amely a run( ) metódust tartalmazza:

```
class SimpleThread {
   public static void main(String[] args) {
      RunnableClass runnableClass = new RunnableClass();
      Thread thread = new Thread(runnableClass);
      thread.start();
   }
}
```

A Thread osztályból példányosított thread objektumot el kell indítani a start( ) speciális metódussal, amely meghívása után elindít egy szálat, és még azelőtt visszatér, hogy az elindított folyamat befejeződne. Ezáltal hosszú számítások esetén semmi nem garantálja, hogy a várt eredmény rendelkezésünkre áll a start( ) metódus visszatértekor.

A szálak prioritása

Ha Windowsban futtatjuk a SimpleThread2 programot, akkor a következő kimenetet kaphatjuk a parancssoron:

```
RunnableClass1 = 0
RunnableClass1 = 1
RunnableClass1 = 2
RunnableClass1 = 3
RunnableClass1 = 4
RunnableClass2 = 0
RunnableClass1 = 5
RunnableClass2 = 1
RunnableClass1 = 6
RunnableClass2 = 2
RunnableClass1 = 7
RunnableClass2 = 3
RunnableClass1 = 8
RunnableClass2 = 4
RunnableClass1 = 9
RunnableClass2 = 5
RunnableClass2 = 6
RunnableClass2 = 7
RunnableClass2 = 8
RunnableClass2 = 9
```

Linuxban és általában a Unix típusú rendszerekben a parancssoron a következő kimenet a valószínű:

```
RunnableClass1 = 0
RunnableClass1 = 1
RunnableClass1 = 2
RunnableClass1 = 3
RunnableClass1 = 4
RunnableClass1 = 5
RunnableClass1 = 6
RunnableClass1 = 7
RunnableClass1 = 8
RunnableClass1 = 9
RunnableClass2 = 0
RunnableClass2 = 1
RunnableClass2 = 2
RunnableClass2 = 3
RunnableClass2 = 4
RunnableClass2 = 5
RunnableClass2 = 6
RunnableClass2 = 7
RunnableClass2 = 8
RunnableClass2 = 9
```

Ennek az eltérésnek az oka, hogy a különböző operációs rendszerek másképpen osztják szét a rendelkezésre álló processzoridőt. A Windows az egyenlősdi híve, ami meg is felel annak a felhasználói rétegnek, amely számára ezt az operációs rendszert tervezték. Az otthoni személyi számítógépen minden program fontos a felhasználó számára, nem érezheti, hogy az egyik lassabban fut, mint a másik. A Unix típusú rendszerekben viszont prioritások vannak, elsődleges szempont például a stabilitás és a biztonság. Minden olyan funkció elsőbbséget élvez, többlet processzoridőhöz jut a többi processz vagy szál rovására.

A prioritás tipikus példája a Linuxban a cron démon által felügyelt lemezellenőrzés, amely olyan fokú előnyt élvez, hogy elindulása után átmenetileg lefagy a rendszer, az internetes kapcsolat halódik. Csak az ellenőrzési feladat végrehajtása után áll vissza a rendszer normális működése. Ez annyira zavaró volt számomra, hogy végül átírtam az /etc/crontab fájlt, egészen más időpontokra ütemezve a mindennapos ellenőrzést.

A Java a Unix típusú ütemezési modellt követi, amit kötött prioritású ütemezésnek nevezünk (fixed priority scheduling). Ennek lényege, hogy a kernel vagy a Java virtuális gép a prioritásuk alapján osztja ki a futtatható szálaknak a processzoridőt, azaz a magasabb prioritásúak hamarabb futnak le, mint az alacsonyabb prioritásúak. De ez nem feltétlenül történik így, esetenként az alacsonyabb prioritású szál is előbbre kerülhet, tehát nincs garancia arra, hogy egy magasabb prioritású szál hamarabb fut le, mint egy alacsonyabb

prioritású. Ugyanakkor a Java ütemezési algoritmus „elővételi jogon" alapul (preemptive), hiszen ha egy magasabb prioritású szál kezd el futni, akkor a többieket háttérbe szorítva azonnal megkezdheti a végrehajtást. A Java szál attól a száltól örökli a prioritását, amelyik elindítja, de a prioritás bármikor megváltoztatható (lásd a SimpleThread3.java forrásfájlt!):

```
thread1.setPriority(java.lang.Thread.MIN_PRIORITY);
thread2.setPriority(java.lang.Thread.MAX_PRIORITY);
```

A prioritás a MIN_PRIORITIY és a MAX_PRIORITY konstansok közötti értéket vehet fel, vagyis 1 és 10 között helyezkedhet el.

Feladat: Futtassuk le a SimpleThread4 programot, és figyeljük meg, hogy a prioritások megváltoztatása hogyan befolyásolja a szálak futását. Célszerű a programot mind Windows, mind pedig valamilyen Unix környezetben kipróbálni, hogy képet kapjunk a két rendszer közti különbségről. Mindenesetre 6 fájlban elmentettem a program kimenetének egy részét a SimpleThread4Output-Linux_x.txt és a SimpleThread4Output-Windows_x.txt fájlokba. A Linux végződésűeket Linuxon, a Windows végződésűeket pedig Windows Milleniumon futtattam. A szálak prioritását a 3. és 4. sor mutatja:

```
thread1 prioritása = 3
thread2 prioritása = 3
```

A SimpleThread4 programban három szál indul, az egyik a fő szál, amely a thread azonosítót kapta, és van két alszál, a thread1 és thread2. A fő szál kapta a legnagyobb prioritást, a két másik szál elsőbbségét mi magunk állíthatjuk be a parancssoron begépelt két szám segítségével:

```
java SimpleThread4 3 7
```

Írjunk be más értékeket is, és figyeljük meg, hogy mi változik. A fő szál a Thread.sleep( ) statikus metódus hatására 3000 milliszekundumig álomba szenderül, majd felébredve ismét magához ragadja a végrehajtást, hiszen „elővásárlási joga" van, maximális prioritást kapott tőlünk. Amikor a fő szál aludni megy, akkor a két másik szál prioritását felcseréljük. Figyeljük meg, hogy ennek milyen a hatása.

Kérdések: Miért veszti el a fő szál az elsőbbségét rögtön indulás után? Mi történik, ha nullánál kisebb, vagy tíznél nagyobb prioritási értéket adunk meg? Küszöböljük ki a hibát. Miért nem tudunk közvetlenül hivatkozni a SimpleThread4 osztály privát elérésű priority01 és priority10 változóira, miért kell a thread.priority01 formát használni? Miért írtuk a thread.priority01 azonosítóban kisbetűvel a thread szót, és miért használtunk nagybetűt a Thread.sleep( ) metódus hívásakor?

A Thread osztály

A java.lang csomagban található egy Thread nevű, nagyon hasznos osztály, amelyben sok statikus, kényelmi metódus van. Ilyen például a fent már említett sleep( ) eljárás, amely az éppen futó szál futását felfüggeszti a paraméterben megadott milliszekundumnak megfelelő időtartamra:

```
public static native void sleep(long millis)
                  throws InterruptedException;
```

Ahogy a metódus lenyomatában látjuk, az egy InterruptedException kivételt dob, amit nekünk egy try-catch blokkal el kell kapnunk:

```
try {
  Thread.sleep(2000);
} catch (InterruptedException e){ }
```

A sleep( ) eljárás túlterhelt, egy másik lenyomatában a milliszekundumok mellett a nanoszekundumokat is megadhatjuk paraméterként:

```
public static void sleep(long millis, int nanos)
                  throws InterruptedException {}
```

Az elaltatott szálat addig nem lehet felébreszteni, amíg a számára kijelölt alvási idő le nem telik. Ekkor viszont felébred, és ha magas a prioritása, akkor azonnal munkába kezd. A yield( ) metódus hasonló funkciót tölt be, mint a sleep( ), mert használatával processzoridőt juttathatunk a többi, alacsonyabb prioritású szálnak:

```
public static native void yield();
```

Ez az eljárás időlegesen felfüggeszti az éppen futó szálat, és lehetőséget ad a többi szálnak a végrehajtásra.

Az isAlive( ) függvénnyel megtudhatjuk, hogy egy szál él-e még. Egy szálat akkor tekintünk élőnek, ha azt már elindítottuk, és még nem halt el:

```
public final native boolean isAlive();
```

A függvény igazként tér vissza, ha a szál él, hamisként, ha nem él. A currentThread( ) függvény az éppen futó szál címét adja vissza. A lenyomata:

```
public static native Thread currentThread();
```

Programjainkban legalább egy szál van, amely meghívja a kitüntetett szerepet játszó, statikus main( ) metódust. Ha semmilyen más szálat nem indítunk el, akkor a currentThread( ) függvény ezt a fő szálat azonosítja:

```
Thread thread = Thread.currentThread();
```

A Thread osztály eddig ismertetett metódusainak lenyomatában szerepel a native kulcsszó, ami azt jelenti, hogy a metódus törzsét nem Java nyelven írták meg, hanem valami más kódban.

A szokásos setter és getter metódusokkal megváltoztathatjuk vagy lekérdezhetjük a szál prioritását:

```
public final void setPriority(int newPriority) {}
public final int getPriority() {}
```

A szálaknak nevet is adhatunk, és azokat lekérdezhetjük:

```
public final void setName(String name) {}
public final String getName() {}
```

A szál nevét a konstruktorokban is átadhatjuk paraméterként, például:

```
public Thread(String name) {}
```

A Thread osztály megvalósítja a Runnable interfészt, ezért őse lehet saját szálainknak, ahogyan azt például a SimpleThread5 csontváz osztályban látjuk:

```
class SimpleThread5 extends Thread {
  public SimpleThread5() {
  }
  public static void main(String[] args) {
  }
  public void run() {
  }
}
```

A Thread osztályból való leszármaztatás a szálak létrehozásának másik módja, ahogy például a SimpleThread4 osztályban látjuk. Ilyenkor a run( ) metódust magában az utód osztály törzsében valósítjuk meg, és nem a konstruktorban adunk át egy futtatható (runnable) külső osztályt. Az ilyen kívülről átadott osztályt cél (target) osztálynak nevezzük, ahogy az a Thread osztály egyik konstruktorának paraméteréből látszik:

```
public Thread(Runnable target) {}
```

Felhívom a figyelmet arra, hogy a paraméterátadás nem osztálynévvel, hanem interfésszel történik. Ez teszi lehetővé, hogy bármilyen osztályt átadhassunk a Thread osztályból létrehozott példánynak, ha az megvalósította a Runnable interfészt.

Látjuk, hogy a szálakkal kapcsolatos osztályok a java.lang csomagba vannak összegyűjtve, tehát az alaprendszer részét

képezik. Ezért nem kell ezt a csomagot importálni a forrásfájlok elején. A Thread osztályban találhatóak még a stop( ), suspend( ) és resume( ) metódusok is, amelyek mára helytelenített API-vá váltak.

A többszálú végrehajtás veszélyei

Logikusnak tűnhet az a feltevés, hogy a start( ) metódussal elindított szálat bármikor leállíthatjuk a stop( ) eljárással, felfüggeszthetjük annak futását a suspend( )-del, majd amikor ismételten szükség van rá, újraindíthatjuk a resume( )-mal. De a tapasztalat mást mutatott. Kiderült, hogy ezek a metódusok nem biztonságosak. A stop( ) meghívása után sérült (damaged) objektumok maradhatnak hátra, és ha más szálak ilyen sérült objektumokhoz fordulnak, akkor kiszámíthatatlan viselkedés lehet az eredmény. A sérülés nem jelentkezik azonnal, órák vagy napok telhetnek el, mire észrevesszük, hogy valami nem úgy működik, ahogyan kellene.

A másik két metódus használata szintén veszélyeket rejt magában, ezek a metódusok szintén lefagyásra hajlamosítók (deadlock-prone). Miután az ilyen típusú hibák napvilágra kerültek, a Javasoft programozói helytelenítetté (deprecated) tették a fenti metódusokat, de nem adtak helyettük más alternatívát. Így a fenti nem javasolt API továbbra is része maradt a Java kódnak, de ha nagyon nem megy másként, kénytelenek vagyunk használni.

Éhező filozófusok

A többszálú programvégrehajtást választó programozóra váró sors illusztrálására egy példabeszédet szoktak idézni. A történet a következő. Egy kerek asztal körül kínai filozófusok gyülekeznek munkaebédre. Mindegyikük előtt van egy tányér rizs, és a tányérok között egy-egy evőpálcika. A filozófusok ragaszkodnak ahhoz, hogy evőpálcikával egyenek, és ha erre nincs módjuk, akár éhen is halnak. A szertartásos étkezés koreográfiája a következő: az egyik filozófus felemeli a tányérja jobb oldalán lévő pálcikát, majd a bal oldalán lévőért nyúl. Ha mindkettő a helyén van, akkor eszik néhány falatot. A mellette lévő filozófus eközben nézi, hogyan eszik a társa, mert neki csak egy evőpálcika jutna, vagy egy sem, hiszen valójában ugyanannyi pálcika van, ahány tányér. Filozófusaink azonban néhány falat után leteszik a kezükben lévő pálcikát, amit így társaik is felvehetnek. Ennek a szertartásrendnek köszönhetően lesznek olyanok, akik hamarabb, míg mások később laknak jól, de a végén mindannyian teli gyomorral távoznak.

Mi történik azonban akkor, ha az összes filozófus egyszerre nyúl a jobb kezénél lévő evőpálcika után, és miután felvette, többé nem hajlandó letenni azt? Ebben az esetben mindegyiküknek egyetlen evőpálcika jut, ami még egy kínai filozófusnak is kevés. Közben hiába várnak arra, hogy felvehessék a másik evőpálcikát, hiszen az a tőlük balra lévő szomszéd jobb kezében van. Ha nagyon makacsok, hosszú éhezés következhet, akár éhen is halhatnak, pedig előttük ott van a tányéron a rizs.

Hasonló a helyzet az olyan szálakkal, amelyek ugyanazokért az erőforrásokért versenyeznek. Ezek működését össze kell hangolni, szinkronizálni kell. Erre való az előző fejezetben már említett synchronized kulcsszó. A Java rendszer automatikusan zárolja az olyan objektumot vagy metódust, amely előtt ez a kulcsszó van, és az ilyen zárolt erőforráshoz másik szál sem férhet hozzá addig, amíg a zárat a Java rendszer (szintén automatikusan) fel nem oldja. Amelyik objektum lekésik az adott erőforrásról, várakozásra kényszerül, és csak a zár feloldása után tudja elvégezni feladatát.

A dolog egyszerű, és nagyszerűen működik is mindaddig, amíg a programozó el nem követ valami hibát. Terjengős és bonyolult programok esetében az ilyen tévedéseknek nagy ára van, mert az ilyen programozási hibák alattomosak, a fejlesztési fázisban csak ritkán vagy egyáltalán nem jelentkeznek, és amikor teszteléskor nyilvánvalóvá válik, hogy valami nem stimmel, akkorra a marketingesek már beharangozták a terméket, nincs idő a hosszas bogarászásra, ami ráadásul növelné is a költségeket. A felhasználókhoz így gyakran hibás termék jut el, emiatt könnyebben rávehetők arra, hogy a következő, immár hibátlannak remélt verziót is megvegyék, hiszen a cég állítása szerint az már nem fog félóránként lefagyni.

Ez az ördögi spirál különösen a zárt forráskódú, fekete doboz technológiákra jellemző, hiszen ott erősen behatároltak a hibajavításra fordítható programozói kapacitások. Az ilyen cégeknél a kínai filozófusok éheznek, esetenként éhen is halnak, az étterem-tulajdonosok viszont szépen gyarapodnak. Ezzel szemben a GNU szellemében folyamatosan fejlesztett és tesztelt Linux ilyen hibáit egy jóval nagyobb programozói közösség hamar kijavítja.

Várakozás és értesítés

Szinkronizációs hibára utalhat, ha olyan abszurd eredményeket kapunk, mint például 2 + 2 = 5. Hasonló zavar keletkezhet, ha például két szál kezdi el kiírni a Ratiosoft szót ugyanoda. A megjelenő szöveg valami ilyesmi lehet: RRatatioiososfttoft. Hangsúlyoznom kell, hogy ez a végeredmény nem determinisztikus, hiszen minden egyes futtatásnál más zagyvalékot kaphatunk. Nehezen felfedezhető hibát okozhat az is, ha egy hosszabb számításnál nem győződünk meg arról, hogy a szál befejezte-e a számításokat, és az eredményt beírta-e az általunk kijelölt változóba. Ha egy ilyen, még fel nem töltött változót olvasunk ki, akkor számításaink megbízhatatlanná válnak, amit első pillantásra talán észre sem veszünk. Hogy ezeket a hibákat elkerüljük, az egyik objektumnak várnia (wait) kell mindaddig, amíg a másik be nem fejezi a működését. Amikor a másik objektum végzett a feladatával, értesítenie (notify) kell erről a várakozó objektumot, vagy minden egyes objektumot (notifyall).

Vegyük a következő példát. Egy dolgozó (Worker) osztály hosszú számításokat végez számunkra, aminek a végeredményét egy tároló (Speicher) osztály típusú objektumba teszi:

```
public class Worker extends Thread {
  public Worker(Speicher speicher) {
      this.speicher = speicher;
  }
  public void run() {
    for (int i = 0; i < 5; i++) {
      randomNumber = Math.random();
      try {
          Thread.sleep(2000);
      } catch (InterruptedException e){ }
      speicher.store(randomNumber);
      System.out.println("Worker store:
            " + randomNumber);
    }
  }
  private double randomNumber;
  private Speicher speicher;
}
```

A hosszú számításokat a Thread.sleep(2000) metódussal imitáljuk, hiszen így a dolgozó osztály a véletlen szám generálása után mindig alszik 2 másodpercet. Miután felébred, eltárolja (store) az eredményt a tároló objektumban, és

tájékoztatásul kiírja azt számunkra a parancssoron. Ugyanakkor létrehozunk egy olyan osztályt, amely a dolgozó osztály által kiszámított eredményre vár. Ezt az osztályt lebzselőnek (Loiterer) fogjuk nevezni, hiszen folyton várakoznia kell, amíg meg nem kapja a kiszámított eredményt:

```
public class Loiterer extends Thread {
  public Loiterer(Speicher speicher) {
      this.speicher = speicher;
  }
  public void run() {
    for (int i = 0; i < 5; i++) {
      double result = speicher.recall();
      System.out.println("Loiterer recall: " +
result);
    }
  }
  private Speicher speicher;
}
```

A lebzselő objektum a tároló objektumból olvassa ki (recall) a double típusú eredményt, majd kiírja azt a parancssorra. Mind a dolgozó, mind pedig a lebzselő osztály a példányosított tároló osztályt használja az eredmény elmentésére és annak kiolvasására:

```
public class Speicher { @MONO =   public synchronized
double recall() {
    while (ready == false) {
      try {
        wait();
      } catch (InterruptedException e) { }
    }
    ready = false;
    notifyAll();
    return result;
  }
  public synchronized void store(double result) {
    while (ready == true) {
      try {
        wait();
      } catch (InterruptedException e) { }
    }
    this.result = result;
    ready = true;
    notifyAll();
  }
  private double result;
  private boolean ready = false;
}
```

Figyeljük meg, hogy csak ebben az osztályban hangoltuk össze a store( ) és a recall( ) metódusok működését a synchronized kulcsszó használatával. Csak ebben az osztályban használtuk a wait( ) és a notifyall( ) metódusokat, amiket minden osztály inherensen örököl a java.lang.Objet osztálytól. A gyorsabb notify( ) eljárást is használhattuk volna, de ennek használata kockázatos olyan esetekben, amikor egyszerre több objektum vár a sorára, mivel ez a metódus közülük csak egyet ébreszt fel, véletlenszerűen. A notifyall( ) eljárás ezzel szemben mindegyik várakozó objektumot értesíti.

A main( ) eljárást tartalmazó Notification osztályban példányosítjuk a Speicher, a Worker valamint a Loiterer osztályokat. A két utóbbi a Thread osztály leszármazottja, és így megvalósítja a Runnable interfészt, és törzsükben megtalálható a kötelező run( ) metódus.

```
public class Notification {
  public static void main(String[] args) {
    Speicher speicher = new Speicher();
    Worker worker = new Worker(speicher);
    Loiterer loiterer = new Loiterer(speicher);
    worker.start();
    loiterer.start();
  }
}
```

Példányosításuk után mindkét szálat elindítjuk a start( ) metódussal. Ezután mindkét szál folyamatosan próbálkozik, hogy a rábízott feladatot elvégezze. Mivel a speicher objektumban a ready változó alapértelmezetten false-ra van állítva, a recall( ) metódust hívó loiterer objektum várakozásra kényszerül. A worker objektum viszont be tudja állítani a result értékét, mivel a hamis ready változó ezt megengedi. Ezután igazra állítja a ready változót, és minden várakozó szálat kiértesít a notifyall( ) metódussal. Ekkor a recall( ) metódus feléled, a ready változót hamisra állítja, értesíti a többi objektumot, és visszaadja a result értékét. A dolgon el kell gondolkodni egy darabig, amíg megérti az ember.

A szinkronizáció következtében a két szál összehangoltan működik, a worker objektum lép elsőként, és a loiterer objektum mindig megvárja. Ezt illusztrálja a következő kimenet:

```
Worker store:     0.002551354039110554
Loiterer recall: 0.002551354039110554
Worker store:     0.5129149946787599
Loiterer recall: 0.5129149946787599
Worker store:     0.27599174052003217
Loiterer recall: 0.27599174052003217
Worker store:     0.4828068424069474
Loiterer recall: 0.4828068424069474
Worker store:     0.6538905896023269
Loiterer recall: 0.6538905896023269
```

Szinkronizáció nélkül a forráskód így nézne ki:

```
public class Speicher_ {
  public double recall() {
    return result;
  }
  public void store(double result) {
    this.result = result;
  }
  private double result;
}
```

A NotificationError program a fenti, szinkronizálatlan metódusokat hívja meg. Ha lefuttatjuk, akkor a fentihez hasonló jó eredményeket is kaphatunk, de hibák is adódhatnak. Az alábbi listázásban azonnal látjuk, hogy a loiterer objektum mindjárt az első lépésben rossz eredménnyel tér vissza:

```
Loiterer recall: 0.0
Worker store:     0.1284091727355431
Loiterer recall: 0.1284091727355431
Worker store:     0.45087809467505136
Loiterer recall: 0.45087809467505136
Worker store:     0.10119549558079155
Loiterer recall: 0.10119549558079155
Worker store:     0.37131351111055866
Loiterer recall: 0.37131351111055866
Worker store:     0.7466783257485079
```

A CD-melléklet output könyvtárába több szövegfájlt tettem, amelyek különböző futtatási eredményeket mutatnak Linux és Windows ME környezetben.

A wait( ) metódus hasonló, mint a sleep( ), de az utóbbinál mindenképpen ki kell várni azt az időt, amennyi ki van jelölve számára, nem lehet idő előtt felébreszteni a szálat. A wait( ) lenyomatában is nagyon hasonló a sleep( )-éhez:

```
public final void wait() throws
                    InterruptedException {}
public final native void wait(long timeout)
                    throws InterruptedException;
public final void wait(long timeout, int nanos)
                    throws InterruptedException {}
```

Figyeljünk arra, hogy a wait( ) csakis egy szinkronizált metódus vagy objektum belsejében hívható meg. Mindig egy ciklusban várakozzunk úgy, ahogyan a fenti példában látjuk, soha ne használjuk az if szerkezetet ilyen célra. Vigyázzunk arra, hogy minden wait( ) metódusnak legyen egy notifyAll( ) párja.

A szálak csoportosítása

A Java rendszer minden szálat beletesz egy szálcsoportba (thread group). Ebben ismét a Unix programozási gyakorlat nyilvánul meg, hiszen ott is processzekről és processzcsoportokról (process groups) beszélünk. A processzcsoport egymással összefüggő processzek együttese, rendszerint összefüggő parancsok sorozata, mint például:

```
ls -l ~ | sort | wc -l
```

Ha mi magunk nem tesszük bele az új szálat egy saját magunk által létrehozott csoportba, akkor a Java rendszer az alapértelmezett csoportba teszi azt, ugyanabban a csoportba, ahol az a szál van, amely az új szálat létrehozta. Ez a szálcsoport a programinduláskor keletkezik, és main a neve. Ha csoportokat hozunk létre, akkor szálak egész csapatát indíthatjuk el újra, vagy állíthatjuk le azokat. A dolog egyetlen szépséghibája, hogy a java.lang.ThreadGroup osztályban rendelkezésünkre álló stop( ), suspend( ) és resume( ) metódusok mára mind helytelenített API-vá váltak:

```
class ThreadGroupExample {
   public static void main(String[] args) {
      ThreadRunnable runnable1 =
                    new ThreadRunnable("runnable1");
      ThreadRunnable runnable2 =
                    new ThreadRunnable("runnable2");
      ThreadRunnable runnable3 =
                    new ThreadRunnable("runnable3");
      ThreadRunnable runnable4 =
                    new ThreadRunnable("runnable4");
      ThreadGroup threadGroup =
                    new ThreadGroup("RatiosoftGroup");
      Thread thread1 = new Thread(threadGroup,
                    runnable1);
      Thread thread2 = new Thread(threadGroup,
                    runnable2);
      Thread thread3 = new Thread(threadGroup,
                    runnable3);
      Thread thread4 = new Thread(runnable4);
      //threadGroup.setMaxPriority(7);
      thread1.start();
      thread2.start();
      thread3.start();
      thread4.start();
      try {
         Thread.sleep(3000);
         System.out.println("Letelt!");
      } catch (InterruptedException e){ }
      threadGroup.stop();
   }
}
```

A fenti ThreadGroupExample példában először létrehozunk 4 olyan objektumpéldányt, melyek megvalósítják a Runnable interfészt, majd egy szálcsoport példányt teremtünk. A thread1, thread2 és thread3 szálak konstruktorainak paraméterként átadjuk a szálcsoport objektumot (threadGroup) és a nekik szánt cél objektumokat. Ez a három szál így a RatiosoftGroup csoportba kerül. Figyeljük meg, hogy a negyedik szálat (thread4) nem adtuk hozzá a csoporthoz, ezért az nem is fog leállni, amikor a threadGroup.stop( ) metódust meghívjuk. A többi három szál futása azonban befejeződik. Ne lepődjünk meg, amikor a ThreadGroupExample.java forrásfájl újrafordításakor figyelmeztetést kapunk, hiszen a helytelenített threadGroup.stop( ) metódust használjuk.

Szálak a Swingben

Kezdetben minden Java könyvtár szálbiztos (thread safe) volt, ami azt jelentette, hogy a programozónak nem kellett szinkronizációval foglalkoznia a Java rendszer függvényeinek meghívásakor, hiszen azok egyidejűleg több szálból is biztonságosan hívhatók voltak. Később a Javasoft programozói feladták ezt az ambiciózus célt, és tudjuk, hogy a Swing nem szálbiztos. Ezzel eleget tettek a szálakkal kapcsolatos legfontosabb szabálynak: Ne használjuk őket! Kétségtelen, hogy bizonyos esetekben nagyon felgyorsítják a programfutást, színesebbé tehetik a programot, de a túlságosan gyakori szinkronizálásnak ára van, és ha túlzásba visszük, akkor többszálasnak szánt programunk a szinkronizáció miatt sokat várakozik, és voltaképpen szekvenciálisan fut le, esetenként pedig le is fagyhat. Az egy szálon futó program biztonságos, és kicsi a valószínűsége, hogy rossz eredményeket kapunk. A többszálas végrehajtás rövid idő alatt nagy teljesítményre képes, de megírása nehézkes és költséges, nehéz kibogarászni a hibákat. Inkább csak akkor használjunk több szálat, ha másképp nem tudjuk megoldani az adott feladatot.

A Swingben az eseménykezelő szál (event-dispatching thread) a legfontosabb, itt hajtódnak végre a különböző cselekmények által generált eseményekhez tartozó kódrészletek, és itt rajzolódnak ki a komponensek. A program a fő szálon (main thread) indul, de a grafikus felület megjelenítése után a végrehajtás átadódik az eseménykezelő szálnak. Az actionPerformed( ) és a komponensek kirajzolását végző paint( ) metódus egyaránt itt köt ki. Ezért van az, hogy programjaink grafikus felülete esetenként lefagy, nem történik újrarajzolás, vagy a komponens nem válaszol az egérkattintásokra. Ilyenkor valami munkaigényes folyamat zajlik a háttérben, ami lefoglalja a rendszer erőforrásait. Alapvető szabály, hogy az eseménykezelő metódusokba írt kód legyen rövid, és nagyon gyorsan végrehajtható, különben a program lelassul, és nem válaszol megfelelően a felhasználói kérésekre. Ha mégis arra kényszerülünk, hogy időigényes feladatot oldjunk meg az eseménykezelő szálban, akkor elgondolkodhatunk azon, hogy elindítsunk-e mégis egy másik szálat.

A következő esetekben célszerű szálakat használni:

— Programindításkor a fő szálon legyen a grafikus felület felépítése, a többi inicializációs feladatot jobb külön szálakban elindítani.

— Célszerű szálakat használni a tipikusan időigényes feladatoknál, amilyen egy hosszú ideig tartó számítás vagy képek és fájlok betöltése.

— Jó szolgálatot tesznek a szálak a meghatározott időnként ismétlődő feladatoknál is.

— Ha más programoktól várunk üzenetet, vagy a hálózaton várunk valami külső igény beérkezésére, erre jobb nem a fő szálat használni.

Ha programjainkban egyáltalán nem használunk szálakat, az nem jelenti azt, hogy nem találkozhatunk a többszálú végrehajtás bonyadalmaival, hiszen az általunk meghívott

Java API-ban található rutinok némelyike többszálú. Erre láthatunk példát, ha elindítjuk a CD-mellékleten található Main osztályt. A bejelentkező kép után megjelenik az Operator Demo című fő ablak, amely a Java virtual machine című alkotásomat jeleníti meg. A Help menüben kattintsanak rá az About... almenüre. Ekkor az About Operator Demo dialógusablakba kerülnek. A Java csésze alatt láthatunk egy Ratiosoft feliratú nyomógombot, amit ha megnyomunk, az átvált About feliratúra. Ilyenkor a HTML panel tartalma megváltozik. Ha elég gyorsan nyomogatjuk ezt a gombot, és szerencsénk van, akkor észrevehetjük, hogy alkalmanként a HTML fájlok tartalma összekeveredik, a panelben egyszerre több fájl szövege jelenik meg, ahogy azt a snapshot könyvtárban lévő pillanatkép04.png képen láthatjuk. A normális állapotot a pillanatkép03.png mutatja. Ennek a kavarodásnak az oka, hogy a HTML fájlokat betöltő Java programrész egy szálon elindítja a fájl betöltését, majd azonnal visszatér. Ha ekkor újabb utasítást kap egy másik fájl betöltésére, akkor azt is elkezdi betölteni. Némi ügyességgel így akár négy-öt fájlt is betölthetünk ugyanabba a panelba. Ha a source könyvtárban lévő forrásfájlokat megnézzük, akkor láthatjuk, hogy a kódban nem található többszálú végrehajtás. Programunk mégis hibásnak bizonyult.

Ha a fenti figyelmeztetések dacára szálakat akarunk indítani programjainkban, akkor a Swingben az invokeLater( ) és az invokeAndWait( ) metódusokat kell segítségül hívni. E kettő funkciója azonos, de az invokeLater( ) eljárás elindítja az új szálat, és azonnal visszatér, míg az invokeAndWait( ) eljárás megvárja a feladat befejeztét. Ez utóbbi megoldás nem ajánlott, mert hosszú időre lefoglalhatja az erőforrásokat. Mindkettő a javax.swing.SwingUtilities osztályban található. Bármelyik szálból indíthatók, de a kívánt kód végül az eseménykezelő szálban fog lefutni. Használata hasonló lesz az eddigiekhez. Először létre kell hozni egy olyan osztályt, amely megvalósítja a Runnable interfészt, az ismert módon be kell írni a végrehajtandó kódrészletet a run( ) metódusba, majd a példányosított futtatható objektumot át kell adni az invoklater( ) függvénynek:

```
Runnable runnable = new Runnable() {
   public void run() {
      //végrehajtandó kód
   }
}
SwingUtilities.invokeLater(runnable);
//SwingUtilities.invokeAndWait(runnable);
```

Nos, ennyit a szálakról. Nem állítom, hogy mindent elmondtam róluk. Csak azt szerettem volna elérni, hogy fogalmat alkossanak a használatukról. Akit alaposabban érdekel a szálak alkalmazása, annak el kell olvasnia egy részletes monográfiát. Így is csak hosszú gyakorlat után alakul ki az a tapasztalat, ami a szálak biztonságos létrehozásához kell, és időnként még utána is tévedhet az ember.

Szaló István
ratiosoft@freemail.hu

eFestival

Egy új kezdeményezés, az „eFestival" szervezőit az a szándék vezette, hogy a digitális környezeteben, multimédiás eszközökkel megvalósított magyar nyelvű alkotásoknak legyen rendszeres bemutatási lehetősége, megmérettetési fóruma. Részt lehet venni rajta eBusiness megoldásoktól kezdve internetes tartalomszolgáltatásokon át a művészi multimédiáig mindazzal, ami a kibontakozóban lévő digitális műfajok produktumaként jelenik meg a webhelyeken, illetve CD és DVD adathordozókon.

Nevezési kategóriák:

— A magyar kulturális örökség digitális megőrzése
— Zene a digitális világban
— Felkészítés az információs társadalomra
— Informatika a demokráciáért
— Mindennapjaink informatikája
— Elektronikus kereskedelem

Pályázni lehet 1999. január 1. után publikált digitális alkotásokkal, illetve a nevezéskor is működő online szolgáltatással. A részletes feltételek megtudhatók a http://www.efestival.hu címen.

Nevezési határidő: 2001. május 2.

Oracle adatbáziskezelés

Egy kézikönyv a kezdőknek, egy a haladóknak

Az Oracle Press és a Panem kiadó két hatalmas kézikönyvet adott ki az Oracle rendszerekről. Az egyik a kezdőkhöz szól, és bevezetést nyújt a dinamikus adatbázis által vezérelt intranetes és webes alkalmazások témakörébe az Oracle8i termékcsalád alapján. A másik kötet teljes referencia az Oracle8 rendszerhez.

A kezdők figyelmébe ajánlott könyv nem tekinthető hagyományos értelemben vett ismertetésnek az Oracle rendszerekről. Csevegő stílusa kellemes olvasmánnyá teszi, de szerkezete túlságosan laza, mondanivalója néhol szétfolyó, és a sok ismétlés olykor már bosszantó, szinte sulykolásnak tűnik. Közben olyan érzésünk támad, hogy sok fontos dolog homályban marad. Keveset tudhatunk meg például az adatelemző rendszerről, az Oracle Discovererről, pedig ez az Oracle döntéstámogató megoldásainak kulcsfontosságú eleme. Az is alig derül ki a szövegből, hogy mire képes az Oracle Application Server.

Elnagyolt rész szól az Oracle Designer modellgeneráló rendszerről. Keveset olvashatunk a többszálas szerverarchitektúráról, bár fontosságát több helyen is kiemelik, mert a Multithreaded Server (MTS) azon eszközök egyike, amelyek lehetővé teszik tízezernél is több (!) felhasználó egyidejű kiszolgálását.

Eltolódott arányok

Nem elég világos a könyvben a termékek alá-fölé rendeltségi viszonya. Ráadásul a verziószámok emelkedésével néhány termék elnevezése is megváltozott. Ami az egyik helyen még Oracle Forms, az újabban már Form Builder.

Nem teljesen egyértelmű, hogy mi mindent sorolnak a szerzők a leglényegesebb fejlesztőeszközök körébe. Az első fejezetben ide tartozik az SQL*Plus, az Oracle Developer, az Oracle Designer és az SQL Loader (az Oracle Developerbe beleértődik az Oracle Forms, az Oracle Reports és az Oracle Graphics is). Másutt, az Oracle Developer tárgyalásakor már Oracle Form Builder és Report Builder szerepel, és mintha a grafikus fejlesztőről szívesen megfeledkeznének.

A könyvből főleg a működtetés oldaláról ismerjük meg a rendszert. Megtaláljuk a telepítés elég részletes leírását, az SQL*Plus kezdő szintű, majd külön fejezetben a haladó szintű ismertetését, és elég sok szó esik a rendszer adatkezelő nyelvéről, a PL/SQL-ről. Két fejezet is foglalkozik az alkalmazáshangolással: az egyik a kezdőknek, a másik a haladóknak. Ugyanez a bemutatás módszere az adatbázis-rendszergazdák tennivalóival kapcsolatban: előbb a feladatok és a megoldásukhoz használható eszközök kerülnek sorra, majd később következik ugyanez a téma haladó szinten. Úgy érzem, hogy ez a fejezet a könyv legtartalmasabb része. Ebből lehet legtöbbet megtudni az egész robusztus szoftverrendszer működéséről, és itt érzékelhetjük a legjobban, hogy milyen módszerekkel érhető el vele a legjobb teljesítmény.

A gyakorlati munka szempontjából rendkívül fontos az a fejezet, amely az adatexport és adatimport módszereinek és eszközeinek ismertetésével foglalkozik. Kitérnek a szerzők a hibakezelés leggyakoribb problémáira is, példákon szemléltetve néhány valós helyzetet. Két varázsló segíti az Oracle Enterprise Manager alatt az export és az import folyamatainak lebonyolítását. Fontos fejezet még a mentés és a visszaállítás ismertetése, különös tekintettel az exportra és az importra. Azt is megtudhatjuk a szövegből, hogy különböző helyzetekben miként végezhető el a rendszer részleges vagy teljes visszaállítása, és hogy milyen módon lehet 24 órás adatbáziselérést biztosítani.

Új koncepció: adattárház

Külön érdemes szólni arról az új fogalomról, amit a könyv szerzői adattárháznak neveznek. A szerzők felfogása szerint az adattárházak olyan döntéshozatali segédeszközök, amelyekkel az adatbázisokban gyűjtött információk összevethetők a központi tárba bekerülő aktuális adatokkal, és az eredményt egyszerű struktúrában lehet megjeleníteni. A döntéshozatal támogatásában a legfontosabb témák:

— Kellő időben rendelkezésre áll-e az információ.

— Mekkora késedelemmel jelenik meg az információ.

— Bemutathatók-e az adatok a szükséges formában.

— Mennyire hihetünk a kapott eredményeknek.

Az Oracle ilyen problémák megoldásához próbál megfelelő eszközöket kidolgozni. Ehhez speciális adatbázisok kifejlesztését találták célszerűnek, amelyek

— célorientáltak, például üzleti folyamatok köré szervezhetők;

— integráltak, tehát kombinálni lehet bennük a célokat (a beszerzést és az eladást például együttesen lehet elemezni);

— az alapvető adatokat tartósan meg tudják őrizni;

— időskálán is mozogni tudnak, tehát nem csak az aktuális adatokról kaphatunk értékelő információkat.

M. Abbey – M. J. Corey – I. Abramson:

Oracle8i

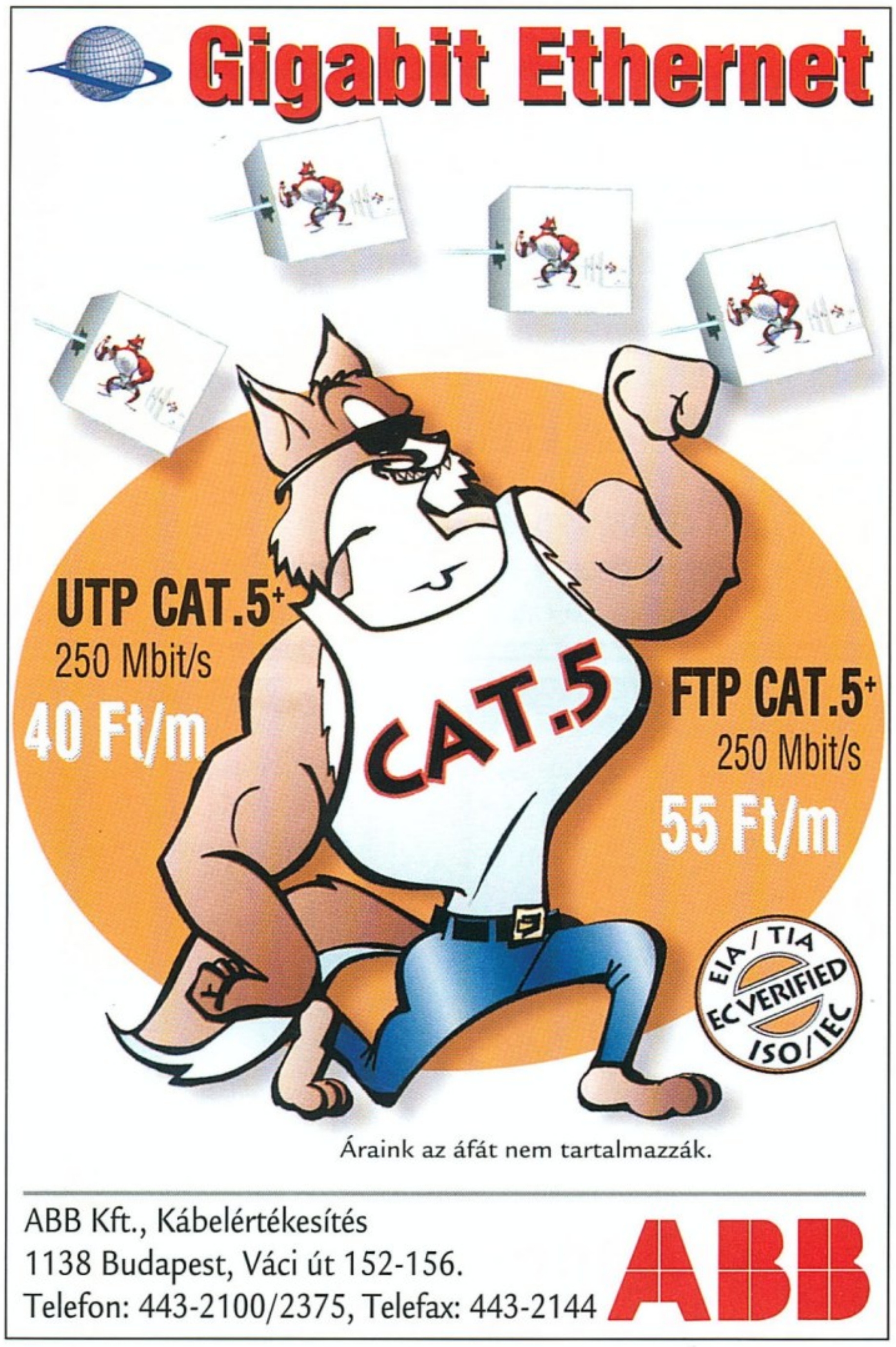
Kézikönyv kezdőknek dinamikus, adatbázisvezérelt intranetes és webes alkalmazásokhoz
Panem – Oracle Press, 2001
624 oldal, 6900 Ft

G. Koch – K. Loney:

Oracle8

Teljes referencia
Panem – Oracle Press, 2001
828 oldal, 5900 Ft


ÚJ CD-GYÁR MAGYARORSZÁGON!
CD-ROM • CD-AUDIO • CD-VIDEO
GYÁRTÁS AKÁR 36 ÓRÁN BELÜL
UTÁNGYÁRTÁS AKÁR 12 ÓRÁN BELÜL!
CD 100 DARABTÓL IS!
KAZETTAGYÁRTÁS TOVÁBBRA IS TÖKÉLETES MINŐSÉGBEN!
MC&CD KFT.
Budaörs, Baross u. 77.
Telefon/fax: (23)416-007
E-mail: mccd@mccd.hu


Gigabit Ethernet
UTP CAT.5+
250 Mbit/s
40 Ft/m
CAT.5
FTP CAT.5+
250 Mbit/s
55 Ft/m
EIA / TIA
EC VERIFIED
ISO / IEC
Áraink az áfát nem tartalmazzák.
ABB Kft., Kábelértékesítés
1138 Budapest, Váci út 152-156.
Telefon: 443-2100/2375, Telefax: 443-2144
ABB


CORG® COMPUTER
CORG COMPUTER KFT.
1111 BARTÓK BÉLA ÚT 46.
TEL.: 466-6675, 381-0135
FAX: 365-6165
Digitális Videó és Audió Centrum
Szinte minden, amire a digitális képalkotástól a feldolgozásig szüksége lehet.
DIGITÁLIS VIDEÓ ESZKÖZÖK
• DV és D8 kamkorderek
Sony, Panasonic, Canon
• DV bemenet kialakítása
a fenti kamkordereknél. Így a megszerkesztett DV anyag formátumkonverzió nélkül visszaírható a DV kazettára. Ezen kívül a D8 kamkordereknél az analóg bemenet is élni fog, megtakarítva ezzel egy analóg bemenetű digitalizáló kártyát.
• DV editáló rendszerek
Canopus – azoknak, akik profi megoldásra törekednek
Pinnacle – Studio DV, DV200, DV500
A nálunk vásárolt DV rendszereknél kedvezménnyel alakítjuk ki a kamkorderek DV bemenetét!
GRAPH ZEPPELIN
DIGITÁLIS FÉNYKÉPEZŐGÉPEK
Végre egy digitális kamera, amivel a profik is elégedettek lehetnek:
• Nikon Coolpix 990
– 3,3 Mpixeles CCD
– 3x autofókusz
– professzionális fénymérési módok
• További kínálatunk
– a NIKON digitális kamerák és scannerek teljes választéka
– állványok, vakuk, kiegészítők
COOLPIX 990
Nikon
http://www.corg.hu


SZOFTVERMÁSOLÁS IPARI MINŐSÉGBEN
NAPI 5000 DB FLOPPY
NAPI 500 DB CD-R SZITÁZVA
max 150MB, szitafilm szükséges
Audio, video és CD-ROM gyártás
Hálózati, automatikus, CD-R/DVDmásoló munkaállomás: COMPOSER
Ipari CD és DVD másolók és nyomtatók: CD/DVD
TETA
TETA MAGNETIC KFT
1131 BP., Rokolya u. 1-13.
T / F : (1) - 350-6773
tetamag@mail.matav.hu

Következtetések

Technikailag a nagy kihívást az jelentette, hogy hatalmas adattömegen kellett hatékonyan végezni a műveleteket. Néhány olyan megállapítás, amely ennek a témakörnek a vizsgálata során kristályosodott ki:

1. A párhuzamosítás és az automatikusan végrehajtható párhuzamos futtatáshangolás sokat segíthet a probléma megoldásában.

2. Több területen is szükség van a párhuzamosításra, így például a kérdésfeldolgozásban, a betöltés során vagy a tábla és az index egyidejű létrehozásában.

3. A párhuzamos kérdésfeldolgozás kulcskérdése a dinamikus egyensúlyi rendszer megteremtése. Ennek lényege, hogy a betöltés dinamikussá tételével a korábban befejeződő feladatok miatt ne kerüljenek hátrányba a lekérdezést végrehajtó folyamatok.

4. Az objektumok particionálására is szükség van. Ezzel (a) kisebb lesz az adatvesztés esélye, (b) elérhető a betöltési egyensúly, (c) jobban megoldható a mentés és a visszaállítás, (d) könnyebb maga az archiválás, a régebbi adatok eltávolítása vagy háttértárba helyezése.

Materializált nézet

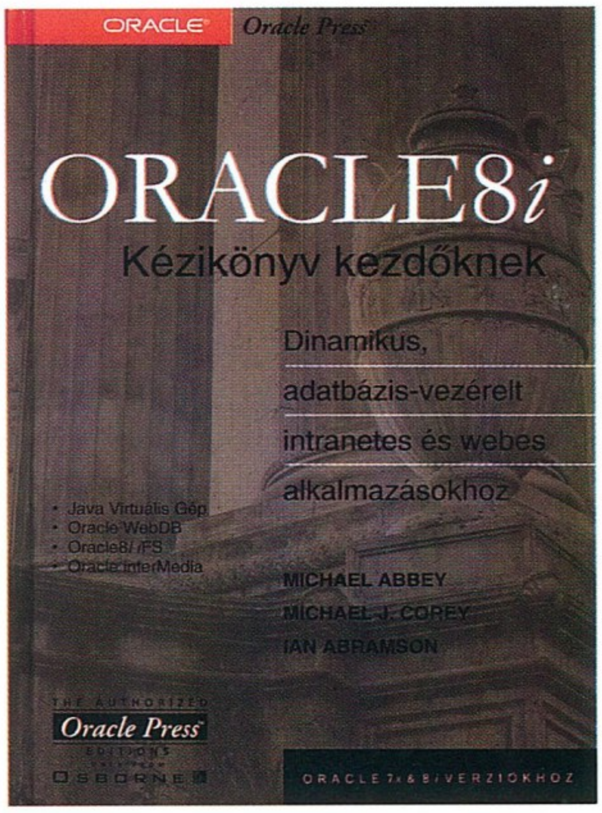
Új objektumokként jelentek meg az adattárházak kidolgozása során a materializált nézetek. A probléma lényege, hogy (1) mikor érdemes előre összegezni és kiszámítani bizonyos eredményeket, és (2) az optimalizáló számára hogyan lehet kezelhetővé tenni ezeket az előre gyártott objektumokat. Mindez az adatbázis használójának a tudta nélkül megy végbe, neki nem kell tudnia, hogy a háttérben a rendszer a maga számára átalakítja, újraírja a lekérdezéseket. A materializált nézetek létrehozásának első lépése az adatok szintjének, az ún. dimenzióknak a meghatározása. A dimenziók olyasféleképpen működnek, ahogyan meg tudjuk tekinteni a képernyőnél nagyobb táblázatok egyes részeit. Keresőtábláknak is nevezhetnénk ezeket a materializált nézeteket, amelyek segítenek automatikusan kiválasztani az éppen szükséges adathalmazt.

Érlelődő problémák

Több olyan módszerről is tájékozódhatnak a könyv olvasói, amelyek az adattárházak fejlődésben lévő elméletének melléktermékei. Általános értelemben is érdeklődésre számot tartó vizsgálati témává izmosodott például a szállítható táblaterületek előállítása és mozgatása.

Ebbe a tágabb feladatkörbe sorolhatjuk például azokat a módszereket, amelyek az aggregációs műveletek körének a kibővítéséből születtek. Az egyik ilyen a különböző szintű összegezések képzése, ami tágítható a részletesség és a teljesség irányába.

Fontos új művelet az aggregáció olyan általánosítása, amely az adatok meghatározott halmazára az összegzések minden lehetséges kombinációját előállítja. Speciális műveletnek tekinthető bizonyos adathalmazok olyan összekapcsolása, ahol egy viszonylag nagy méretű tábla adatait kell egybevetni több kisebb tábla adataival. Ezek a „csillagsémaszerűen" elvégezhető műveletek olyankor válnak szükségessé, amikor az optimalizálást vezénylő program felismeri, hogy (1) az egyik tábla sokkal nagyobb a többinél, és (2) létezik olyan megszorítás, amely kulcsok segítségével valami hasonló helyzetet állít elő.



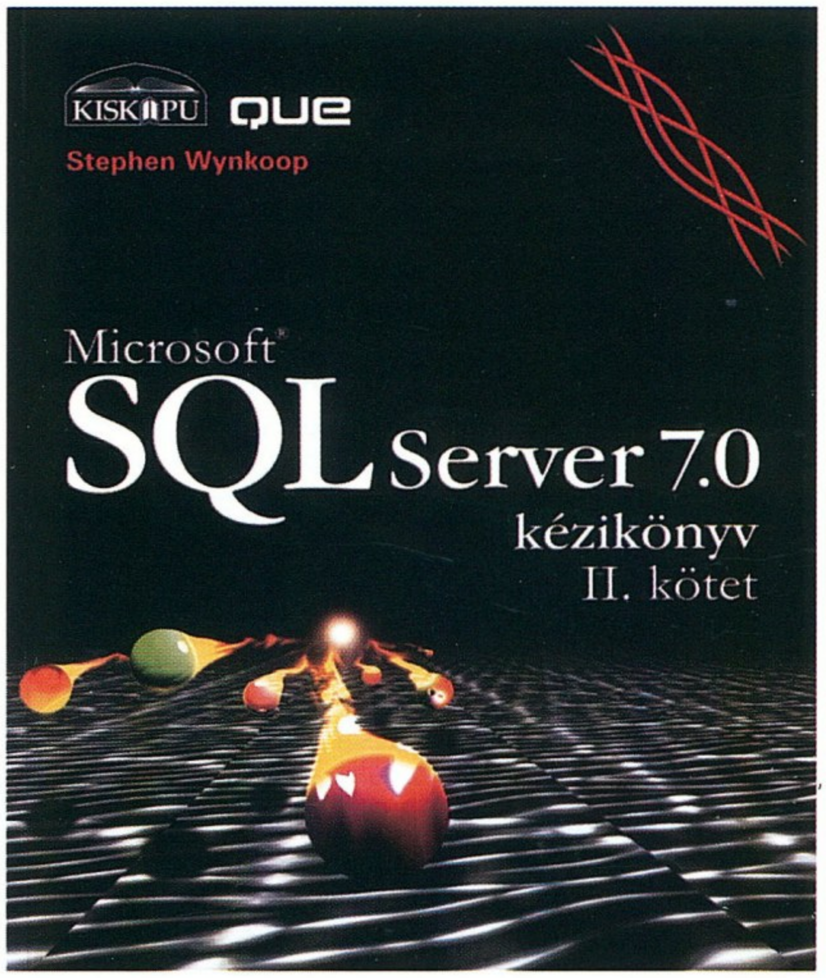
Summázatként megállapítható, hogy sok érdekes probléma lapul a könyvben, csak kifejtésük kissé aránytalan. A magam részéről kiszórnám a felesleges ismétléseket, és szívesebben venném, ha a szerzők a kezdőknek is pontosan meghatároznák mindazt, ami szóba kerül. A mondanivaló és a szaktudás bőven megvan hozzá, a téma jelentősége pedig indokolttá tenné ezt, ráadásul a kötetek ára is meglehetősen borsos.

Vargha Dénes

Nyelv + alkalmazás

Microsoft SQL Server 7.0

Az adatbázisok kezelése az idők folyamán sokat változott, és a számítógépes adatbáziskezelés is nagy utat tett meg az egy gépen futó interpreteres megoldásoktól (amilyen a dBase volt), az adatbázisszerverekre alapozott, dedikált rendszerekkel operáló kiszolgálókig. Ez utóbbi a jelenlegi információtechnológia felhasználásával alaposan kitágítja azokat a lehetőségeket is, amelyekkel a szoftverfejlesztők élhetnek, amikor az adatbáziskezelés háttériparát építgetik.



A Microsoft adatbáziskezelő eszköztárában az xBase, pontosabban annak egyik nyelvjárása, a FoxBase az alapja a Visual FoxPro különböző verzióinak. A termék valószínűleg a Windows 3.x korszakban élte fénykorát, bár elkészült annak új, 32 bites verziója is. Ez azonban az Office csomagban lévő Access-hez hasonlóan nem alkalmas a nagyobb adatbázisokkal végzett munkára, inkább a házi adatbáziskezelés eszköztárába sorolható.

A vállalati adatbáziskezeléshez fejlesztette ki a Microsoft SQL szerverét, amellyel már elég régóta dolgoznak a relációs adatbázisokkal foglalkozók. A Windows NT-re alapozott hálózatokban igazán elterjedt verziója (különösen Magyarországon) a Microsoft SQL Server 6.5 lett, amelyhez számos javítócsomagot adtak ki.

Tény, hogy ezt a két programot korántsem érte annyi kritika, mint a Microsoft asztali oprendszereit és azok holdudvarát. Nem véletlen tehát, hogy amikor Redmondban bejelentették az SQL Server új, 7-es verzióját, annak publikus béta-verzióját szívesen beletettük lapunkba, az 1998/8. szám extra CD-mellékleteként. Azért is előzte meg nagy várakozás ezt a programot, mert eleinte úgy tűnt, hogy szinte egyidejűleg jelenik meg a Windows NT 5-tel, amely végül nagy késéssel, Windows 2000-ként lépett a porondra.

Miként az SQL mint adatbázis-lekérdező nyelv (Stuctured Query Language) nem tartozik az autodidakta módszerrel könnyen megtanulhatóak közé, a számítástechnikai rendszerekben történő implementáció kiismerése sem az. A teljes Microsoft SQL Server csomag képességeit és lehetőségeit alaposan át kell tanulmányozni ahhoz, hogy értő módon tudjuk alkalmazni a szoftvert. Ehhez nyújt segítséget egy jó kézikönyv.

Azt mindenképpen érdemes előrebocsátani, hogy olyan kézikönyv nem igazán készíthető, amely minden lehetséges probléma megoldásában segíteni tud. Különösen érvényes ez a nagy adatbázisrendszereket kezelő alkalmazásokra, de éppen ezeknél nagyon fontos, hogy biztos alapokra lehessen támaszkodni.

A Kiskapu Kft kiadásában, Stephen Wynkoop tollából megjelent kézikönyv sokat segíthet a Microsoft SQL Server 7.0 megismerésében. A két kötetben, de folyamatos oldalszámozással megjelent mű a kézikönyvektől elvárható módon vezet végig az új adatbáziskezelőn, és szerencsére azokon a sajátosságokon is, amelyek ismerete alapszinten sem nélkülözhető az új verzió működtetéséhez. Bemutatja a szabványos ANSI SQL-t, és tartalmaz néhány olyan témakört, amely az alkalmazhatóságot sokoldalúbbá teszi. Ilyen például a memória és a lemezterület dinamikus foglalása, a korábbi külső fejlesztések kiváltása, vagy a hatékonyabb elemzést szolgáló beépített OLAP (Online Analytical Processing).

Ezek természetesen csak kiragadott példák, hiszen igen terjedelmes műről van szó. Maga a téma sem tartozik a legkönnyebbek közé, és a könyv elsősorban azokhoz szól, akiket a mindennapi munka bonyodalmain kell átsegíteni. Szép számban közölnek a könyvben képernyőképeket is, bár azok mérete nem alkalmas arra, hogy a rajtuk lévő információkat könnyedén (nagyító nélkül) elolvassuk. Kár, mert magának a könyvnek jól olvasható a szövege.

A könyvhöz mellékelt CD tulajdonképpen komplett segédletgyűjtemény, amelyből a programozási feladatokhoz, az ismeretgyűjtéshez, a rendszer könnyebb használatához kapunk segítséget. Talán jó lett volna legalább a keretprogramban fel-felbukkanó szövegeket honosítani, bár az ennek segítségével települő részek amúgy sem magyarul szólalnak meg.

Simay Endre István

Stephen Wynkoop:

Microsoft SQL Server 7.0

Kézikönyv I-II.
Kiskapu Kft, Budapest, 2000
898 oldal, 8960 Ft

ELŐFIZETÉS

Az 2001/......... számtól kezdődően előfizetem

az Új Alaplap című CD-mellékletes havi számítástechnikai szaklapot

...... példányban ❑ 1 évre ❑ 1/2 évre

Az éves előfizetési díj: **8960 Ft** (áfával együtt)

❑ **Számlát** kérek (banki átutalással fizetek) ❑ **Befizetési csekket** kérek

Név: ....

(Cég:) ....

Cím: ....

Irányítószám, helység: ....

Dátum:

/aláírás/

ELŐFIZETÉS DIÁKKEDVEZMÉNNYEL

Az 2001/......... számtól kezdődően előfizetem

az Új Alaplap című CD-mellékletes havi számítástechnikai szaklapot

...... példányban ❑ 1 évre ❑ 1/2 évre

Az éves előfizetési díj **diákkedvezménnyel: 8000 Ft** (áfával együtt)

❑ **Számlát** kérek (banki átutalással fizetek) ❑ **Befizetési csekket** kérek

Név: ....

Iskola: ....

Cím: ....

Irányítószám, helység: ....

Dátum: **Az oktatási intézmény igazolása:**

KORÁBBI SZÁMOK MEGRENDELÉSE

Az Alaplap / Új Alaplap korábbi számai közül megrendelem **postai utánvéttel** az alábbiakat:

....

A **3 hónapnál régebbi** CD-mellékletes számok ára 400 Ft, a floppymellékleteseké 200 Ft, a 3 hónapnál frissebb számok teljes árúak. (Az árak áfával együtt értendők, plusz a mindenkori postaköltség.)

Név: ....

(Cég:) ....

Cím: ....

Irányítószám, helység: ....

Dátum:

/aláírás/

Belföldön
díjmentesen is
feladható

ÚJ ALAPLAP

VI., Dózsa György út 84/b
Postafiók 571

1539 Budapest

Belföldön
díjmentesen is
feladható

ÚJ ALAPLAP

VI., Dózsa György út 84/b
Postafiók 571

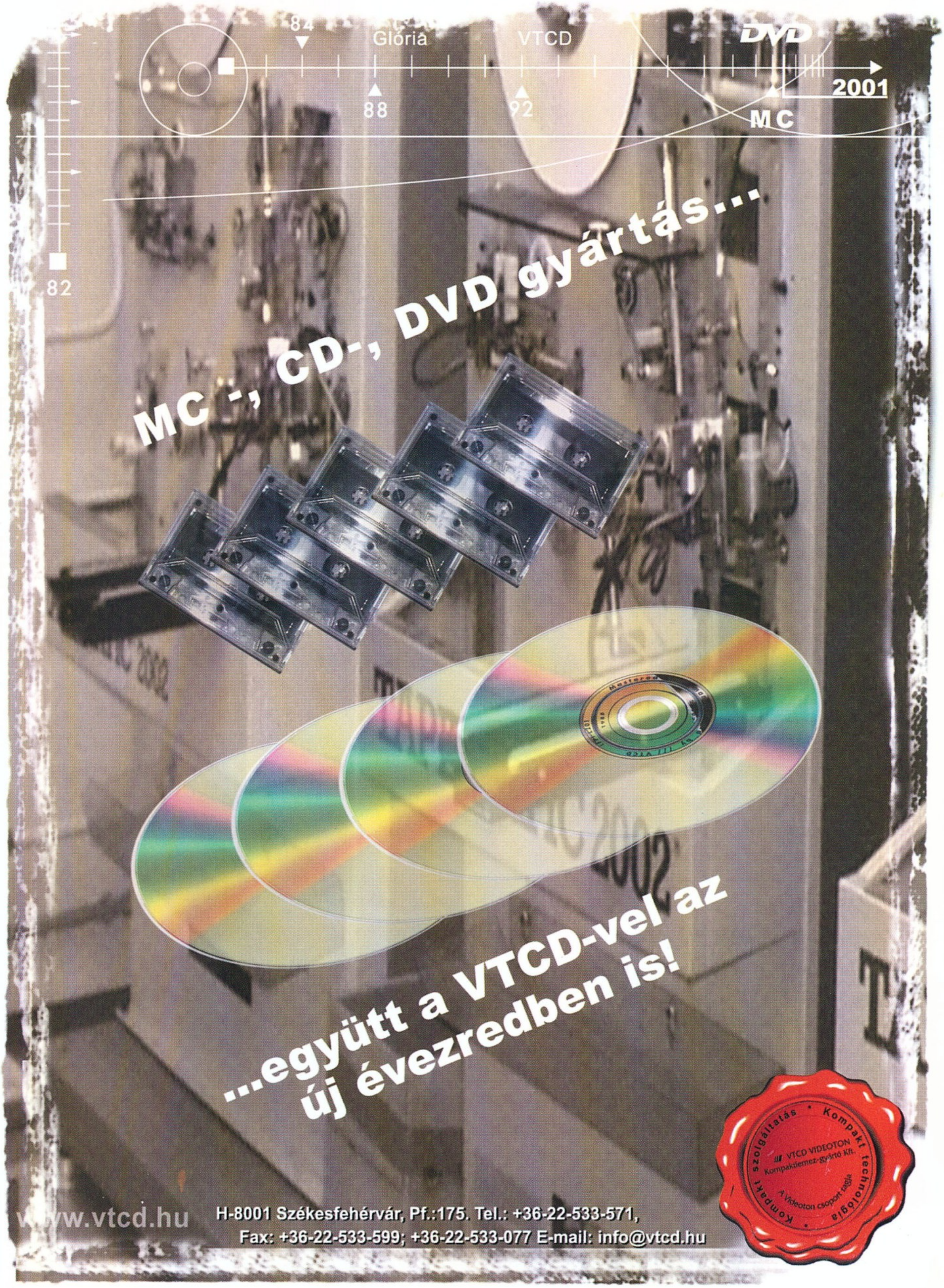
1539 Budapest

Belföldön
díjmentesen
feladható

ÚJ ALAPLAP

VI., Dózsa György út 84/b
Postafiók 571

1539 Budapest



84
Glória
VTCD
DVD
88
92
2001
MC
82
MC -, CD-, DVD gyártás...
...együtt a VTCD-vel az új évezredben is!
Kompakt szolgáltatás • Kompakt technológia •
VTCD VIDEOTON Kompaktlemez-gyártó Kft.
A Videoton csoport tagja
www.vtcd.hu
H-8001 Székesfehérvár, Pf.:175. Tel.: +36-22-533-571,
Fax: +36-22-533-599; +36-22-533-077 E-mail: info@vtcd.hu